

HPSS

Installation Guide

*High Performance Storage System
Release 6.2*

July 2008 (Revision 2.0)

© Copyright (C) 1992, 2008 International Business Machines Corporation, The Regents of the University of California, Los Alamos National Security, LLC, Lawrence Livermore National Security, LLC, Sandia Corporation, and UT-Battelle.

All rights reserved.

Portions of this work were produced by Lawrence Livermore National Security, LLC, Lawrence Livermore National Laboratory (LLNL) under Contract No. DE-AC52-07NA27344 with the U.S. Department of Energy (DOE); by the University of California, Lawrence Berkeley National Laboratory (LBNL) under Contract No. DE-AC02-05CH11231 with DOE; by Los Alamos National Security, LLC, Los Alamos National Laboratory (LANL) under Contract No. DE-AC52-06NA25396 with DOE; by Sandia Corporation, Sandia National Laboratories (SNL) under Contract No. DE-AC04-94AL85000 with DOE; and by UT-Battelle, Oak Ridge National Laboratory (ORNL) under Contract No. DE-AC05-00OR22725 with DOE. The U.S. Government has certain reserved rights under its prime contracts with the Laboratories.

DISCLAIMER

Portions of this software were sponsored by an agency of the United States Government. Neither the United States, DOE, The Regents of the University of California, Los Alamos National Security, LLC, Lawrence Livermore National Security, LLC, Sandia Corporation, UT-Battelle, nor any of their employees, makes any warranty, express or implied, or assumes any liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights .

Printed in the United States of America.

HPSS Release 6.2
July 2008 (Revision 2.0)

High Performance Storage System is a trademark of International Business Machines Corporation.
IBM is a registered trademark of International Business Machines Corporation.
IBM, DB2, DB2 Universal Database, AIX, pSeries, and xSeries are trademarks or registered trademarks of International Business Machines Corporation.
AIX and RISC/6000 are trademarks of International Business Machines Corporation.
UNIX is a registered trademark of the Open Group.
Linux is a registered trademark of Linus Torvalds in the United States and other countries.
Kerberos is a trademark of the Massachusetts Institute of Technology.
Java is a registered trademark of Sun Microsystems, Incorporated in the United States and other countries.
ACSL is a trademark of Sun Microsystems, Incorporated.
Microsoft Windows is a registered trademark of Microsoft Corporation.
NFS, Network File System, and ACSL are trademarks of Sun Microsystems, Inc.
DST is a trademark of Ampex Systems Corporation.
Other brands and product names appearing herein may be trademarks or registered trademarks of third parties.

Table of Contents

Chapter 1. Release 6.2	15
1.1.New Features.....	15
1.1.1.DCE Replacement	15
1.1.2.Linux Support	15
1.1.3.Security.....	15
1.1.4.SCSI PVR	15
1.1.5.HPSS VFS Interface.....	15
1.1.6.GridFTP Interface.....	15
1.1.7.SAN3P/PIO Support	15
1.1.8.Additional hpssadm Configuration Options.....	15
1.1.9.Additional hpssadm operations.....	16
1.1.10.Additional Library and Device Support.....	16
1.1.11.SAN Virtual Volume ID Mapping.....	16
1.1.12.Drive Pools.....	17
1.1.13.FTP Enhancement.....	17
1.1.14.mkhpss Enhancement.....	17
1.1.15.DB2 Monitoring.....	17
1.1.16.File Family enhancements.....	17
1.1.17.SSM Configuration Files Consolidation.....	17
1.1.18. Mover Enhancement.....	18
1.2.Retired Features	18
1.3.Deferred Features.....	18
1.4.HPSS Changes	18
1.4.1.Documentation Organization Changes.....	18
1.4.2.Metadata Changes.....	18
1.4.3.DMAP Gateway Changes	19
1.4.3.1.Creating an XDSM Fileset.....	19
1.4.3.2.Viewing DMAP Gateway XDSM Fileset Information.....	20
1.4.3.3.Viewing Core Server XDSM Fileset Information.....	20
1.4.4.SSM Changes.....	21
1.4.4.1.Changes Affecting Sites Upgrading Directly from 4.5.....	21
1.4.4.2.Changes Affecting Sites Upgrading from 5.1.....	27
Chapter 2. HPSS Basics	35
2.1.Introduction.....	35
2.2.HPSS Capabilities.....	35
2.2.1.Network-centered Architecture.....	35
2.2.2.High Data Transfer Rate.....	35
2.2.3.Parallel Operation.....	35
2.2.4.Based on Standard Components.....	36
2.2.5.Data Integrity Through Transaction Management.....	36
2.2.6.Multiple Hierarchies and Classes of Services.....	36
2.2.7.Storage Subsystems.....	36
2.3.HPSS Components.....	37
2.3.1.HPSS Files, Filesets, Volumes, Storage Segments and Related Metadata	38
2.3.2.HPSS Servers.....	40

2.3.3.HPSS Storage Subsystems.....	44
2.3.4.HPSS Infrastructure	44
2.3.5.HPSS User Interfaces	46
2.3.6.HPSS Management Interfaces.....	46
2.3.7.HPSS Policy Modules	47
2.4.HPSS Hardware Platforms.....	48
2.4.1.Server Platforms	48
2.4.2.Client Platforms	48
2.4.3.Mover Platforms.....	49
Chapter 3. HPSS Planning.....	51
3.1.Overview.....	51
3.1.1.HPSS System Architecture.....	51
3.1.2.HPSS Configuration Planning.....	52
3.1.3.Purchasing Hardware and Software.....	54
3.1.4.HPSS Operational Planning.....	55
3.1.5.HPSS Deployment Planning.....	55
3.2.Requirements and Intended Uses for HPSS.....	56
3.2.1.Storage System Capacity.....	56
3.2.2.Required Throughputs.....	56
3.2.3.Load Characterization.....	56
3.2.4.Usage Trends.....	56
3.2.5.Duplicate File Policy.....	57
3.2.6.Charging Policy.....	57
3.2.7.Security.....	57
3.2.7.1.Cross Realm Access.....	57
3.2.8.High Availability Option.....	58
3.3.Prerequisite Software Considerations	58
3.3.1.Prerequisite Software Overview.....	58
3.3.1.1.DB2.....	58
3.3.1.2.Kerberos.....	58
3.3.1.3.LDAP and IBM Kerberos.....	59
3.3.1.4.Java.....	59
3.3.2.Prerequisite Summary By HPSS Node Type.....	59
3.3.2.1.HPSS Server Nodes.....	59
3.3.2.1.1.AIX Requirements.....	59
3.3.2.1.2.Linux Requirements.....	60
3.3.2.2.HPSS Mover Nodes	60
3.3.2.2.1.AIX Requirements.....	60
3.3.2.2.2.Linux Requirements.....	60
3.3.2.2.3.Solaris Requirements.....	61
3.3.2.2.4.IRIX Requirements.....	61
3.3.2.3.HPSS Client Nodes.....	61
3.3.2.3.1.SSM Client Requirements.....	61
3.3.2.3.2.Client API Requirements.....	61
3.3.2.3.3.FTP/PFTP Client Requirements.....	61
3.3.2.4.HPSS HDM Nodes (Linux only).....	61
3.4.Hardware Considerations	62
3.4.1.Network Considerations.....	62
3.4.2.Robotically Mounted Tape.....	62
3.4.2.1.IBM 3494.....	63
3.4.2.2.Drive-Controlled LTO Libraries (IBM 3582, IBM 3583, IBM 3584, Spectralogic T120).....	63

3.4.2.3.	<i>STK L40, STK SL500, STK SL8500</i>	63
3.4.2.4.	<i>STK</i>	63
3.4.2.5.	<i>ADIC AML</i>	63
3.4.3.	Manually Mounted Tape.....	63
3.4.4.	Tape Devices.....	63
3.4.4.1.	<i>Multiple Media Support</i>	64
3.4.5.	Disk Devices.....	67
3.4.6.	Special Bid Considerations.....	68
3.5.	HPSS Sizing Considerations.....	68
3.5.1.	HPSS User Storage Space.....	69
3.5.2.	HPSS Infrastructure Storage Space.....	69
3.5.3.	HPSS Filesystems.....	71
3.5.3.1.	<i>/opt/hpss</i>	71
3.5.3.2.	<i>/var/hpss</i>	71
3.5.3.3.	<i>/var/hpss/adm/core</i>	72
3.5.3.4.	<i>/var/hpss/hpssdb</i>	72
3.5.3.5.	<i>/var/hpss/hpssdb/subsys1 & subsysX</i>	72
3.5.3.6.	<i>/db2/backups/cfg</i>	72
3.5.3.7.	<i>/db2/backups/subsys1 & subsysX</i>	73
3.5.3.8.	<i>/db2/log/cfg</i>	73
3.5.3.9.	<i>/db2/log/subsys1 & subsysX</i>	73
3.5.3.10.	<i>/db2/mirror-log/cfg</i>	73
3.5.3.11.	<i>/db2/mirror-log/subsys1 & subsysX</i>	73
3.5.3.12.	<i>/db2/mirror-backup/cfg</i>	73
3.5.3.13.	<i>/db2/mirror-backup/subsys1 & subsysX</i>	73
3.5.3.14.	<i>SUBSYS1 Database Allocation</i>	73
3.5.4.	HPSS Metadata Space	74
3.5.4.1.	<i>SMS versus DMS Space</i>	74
3.5.4.2.	<i>'CFG' Database Allocation</i>	74
3.5.4.3.	<i>'SUBSYS' Database Allocation</i>	74
3.5.4.4.	<i>DB2 Disk Space</i>	77
3.5.5.	System Memory and Disk Space.....	78
3.5.5.1.	<i>Operating System Disk Spaces</i>	78
3.5.5.2.	<i>System Disk Space Requirements for Running SSM</i>	78
3.5.5.3.	<i>System Memory and Paging Space Requirements</i>	78
3.6.	HPSS Interface Considerations.....	79
3.6.1.	Client API	79
3.6.2.	FTP.....	79
3.6.3.	Parallel FTP.....	80
3.6.4.	XFS.....	80
3.7.	HPSS Server Considerations.....	80
3.7.1.	Core Server.....	81
3.7.2.	Migration/Purge Server.....	83
3.7.3.	Gatekeeper.....	84
3.7.4.	Location Server	86
3.7.5.	PVL.....	86
3.7.6.	PVR.....	86
3.7.6.1.	<i>STK PVR</i>	87
3.7.6.2.	<i>LTO PVR</i>	87
3.7.6.3.	<i>3494 PVR</i>	88
3.7.6.4.	<i>AML PVR</i>	88
3.7.6.5.	<i>Operator PVR</i>	88

3.7.6.6.SCSI PVR.....	88
3.7.7.Mover	89
3.7.7.1.AIX Asynchronous I/O.....	89
3.7.7.2.Tape Devices.....	89
3.7.7.2.1.AIX.....	89
3.7.7.2.2.Solaris.....	89
3.7.7.2.3.IRIX.....	90
3.7.7.2.4.Linux.....	90
3.7.7.3.Disk Devices.....	90
3.7.7.4.Performance.....	91
3.7.8.Logging Service.....	91
3.7.9.Startup Daemon.....	92
3.7.10.Storage System Management.....	92
3.8.Storage Subsystem Considerations.....	94
3.9.Storage Policy Considerations	94
3.9.1.Migration Policy	94
3.9.1.1.Migration Policy for Disk.....	94
3.9.1.2.Migration Policy for Tape.....	95
3.9.2.Purge Policy	95
3.9.3.Accounting Policy and Validation	96
3.9.4.Security Policy.....	98
3.9.4.1.Client API.....	98
3.9.4.2.FTP/PFTP.....	98
3.9.4.3.XFS.....	98
3.9.4.4.Name Space.....	98
3.9.4.5.Security Audit.....	99
3.9.5.Logging Policy.....	99
3.9.6.Location Policy	99
3.9.7.Gatekeeping.....	99
3.10.Storage Characteristics Considerations	101
3.10.1.Storage Class.....	102
3.10.1.1.Media Block Size Selection.....	103
3.10.1.2.Virtual Volume Block Size Selection (disk).....	103
3.10.1.3.Virtual Volume Block Size Selection (tape).....	103
3.10.1.4.Stripe Width Selection.....	103
3.10.1.5.Blocks Between Tape Marks Selection (tape only).....	104
3.10.1.6.Minimum Storage Segment Size Selection (disk only).....	105
3.10.1.7.Maximum Storage Segment Size Selection (disk only).....	105
3.10.1.8.Maximum VVs to Write (tape only).....	106
3.10.1.9.Average Number of Storage Segments (disk only).....	106
3.10.1.10.PV Estimated Size / PV Size Selection.....	106
3.10.1.11.Optimum Access Size Selection.....	106
3.10.1.12.Some Recommended Parameter Values for Supported Storage Media.....	106
3.10.1.12.1.Disk Media Parameters.....	107
3.10.1.12.2.Tape Media Parameters.....	107
3.10.2.Storage Hierarchy.....	109
3.10.3.Class of Service.....	110
3.10.3.1.Selecting Minimum File Size.....	110
3.10.3.2.Selecting Maximum File Size.....	110
3.10.3.3.Selecting Stage Code.....	110
3.10.3.4.Selecting Optimum Access Size.....	111
3.10.3.5.Selecting Average Latency.....	111
3.10.3.6.Selecting Transfer Rate.....	112
3.10.3.7.StripeLength and StripeWidth Hints.....	112

3.10.4.File Families.....	112
3.11.HPSS Performance Considerations.....	112
3.11.1.DB2.....	112
3.11.2.Bypassing Potential Bottlenecks	113
3.11.3.Configuration.....	113
3.11.4.FTP/PFTP.....	114
3.11.5.Client API.....	115
3.11.6.Core Server	115
3.11.7.Location Server.....	115
3.11.8.Logging.....	115
3.11.9.Cross Realm Trust.....	115
3.11.10.Gatekeeping.....	115
3.11.11.XFS	116
3.11.12.HPSS VFS Interface.....	116
3.12.HPSS Metadata Backup Considerations	117
3.13.HPSS Security Considerations.....	117
Chapter 4. System Preparation.....	119
4.1.General Setup.....	119
4.2.Setup Filesystems.....	120
4.2.1.DB2 Filesystem.....	120
4.2.2.HPSS Filesystem.....	121
4.3.Setup Tape Libraries.....	121
4.3.1.Special LTO Considerations.....	121
4.3.2.IBM 3584.....	121
4.3.3.3494.....	122
4.3.4.STK.....	123
4.3.5.AML.....	123
4.4.Verify Tape Drives.....	124
4.4.1.AIX.....	124
4.4.2.Solaris.....	125
4.4.3.IRIX.....	126
4.4.4.Linux.....	126
4.5.Setup Disk Drives.....	126
4.5.1.AIX.....	127
4.5.2.Linux.....	128
4.5.3.IRIX.....	128
4.6.Setup Network Parameters.....	129
4.6.1.HPSS.conf Configuration File.....	132
4.6.2.SP/x Switch Device Buffer Driver Buffer Pools.....	133
Chapter 5. HPSS Installation and Infrastructure Configuration.....	135
5.1.Prepare for Installation.....	135
5.1.1.Distribution Media.....	135
5.1.2.Software Installation Packages.....	135
5.1.3.Create Owner Account for HPSS Files.....	136

5.1.4. Installation Target Directory Preparation.....	136
5.2. Install Prerequisite Software.....	137
5.2.1. Install Java.....	137
5.2.2. Install MIT Kerberos (If Using Kerberos Authentication).....	137
5.2.3. Install LDAP (If Using LDAP Authorization).....	137
5.2.4. Install Prerequisite Software for XFS HDM.....	138
5.3. Install HPSS/DB2 and Configure HPSS Infrastructure.....	139
5.3.1. Install and Configure HPSS - Root Subsystem Machine.....	139
5.3.1.1. Pre-Installation Configuration.....	139
5.3.1.2. Install HPSS Documentation and DB2 Software.....	141
5.3.1.3. Set Up DB2 Permanent License.....	142
5.3.1.4. Configure HPSS Security Services	143
5.3.1.4.1. Configure UNIX Authentication and UNIX Authorization.....	143
5.3.1.4.2. Configure Kerberos Authentication and UNIX Authorization.....	146
5.3.1.4.3. Configure Kerberos Authentication and LDAP Authorization.....	149
5.3.1.5. Configure DB2 Services.....	152
5.3.1.5.1. Remote DB2 Client Access & Fileset Creation/Deletion.....	157
5.3.1.6. Configure Other Services.....	158
5.3.1.7. Create Configuration Bundle.....	159
5.3.2. Install and Configure HPSS – Secondary Subsystem Machine.....	160
5.3.2.1. Pre-Installation Configuration.....	160
5.3.2.2. Install HPSS Documentation and DB2 Software on a subsystem.....	161
5.3.2.3. Set Up DB2 Permanent License.....	162
5.3.2.4. Install Configuration Bundle.....	163
5.3.2.5. Configure HPSS Security Services	164
5.3.2.6. Configure DB2 Services.....	165
5.3.2.7. Configure Other Services.....	167
5.3.3. Install and Configure HPSS – Mover/Client Machine.....	168
5.3.3.1. Install Mover/Client source code.....	168
5.3.3.2. Install Configuration Bundle.....	169
5.3.3.3. Create /var/hpss subdirectories.....	169
5.3.3.4. Modify Kerberos Configuration File, If Necessary.....	169
5.3.3.5. Check Time and IP Address.....	169
5.4. Post Installation Procedures.....	169
5.5. HPSS Documentation & Manual Page Setup.....	170
5.5.1. Documentation and SSM Help Package.....	170
5.5.2. Manual Page Setup.....	171
5.6. Define HPSS Environment Variables.....	171
5.7. Tune DB2.....	171
5.8. Install and Build HPSS Source Code.....	172
5.8.1. Construct and Build the HPSS Base Source Tree	172
5.8.1.1. Construct the HPSS Source Tree.....	172
5.8.1.2. Build the HPSS Base Source Tree.....	172
5.8.1.3. Generate and Bind the DB2 Helper Program.....	173
5.8.2. Construct and Build the HPSS Mover/Client Source Tree.....	174
5.8.2.1. Construct the HPSS Mover/Client Source Tree.....	174
5.8.2.2. Build the HPSS Mover/Client Source Tree.....	174
5.8.3. Construct and Build the HPSS HDM Source Tree.....	175
5.8.3.1. Construct the HPSS HDM Source Tree.....	175
5.8.3.2. Build the HPSS HDM Source Tree.....	175
5.9. Supporting Both Unix and Kerberos Authentication for SSM.....	175

Chapter 6. Upgrading to HPSS Release 6.2	179
6.1.Special Instructions for Upgrading to HPSS 6.2.2.....	179
6.2.Planning for the HPSS 6.2 Upgrade.....	180
6.2.1.Metadata changes in HPSS 6.2.....	180
6.2.2.Upgrade Requirements and Limitations.....	182
6.2.3.New Authentication and Authorization Mechanisms.....	182
6.2.3.1.Authentication Mechanisms.....	183
6.2.3.2.Authorization Mechanisms.....	183
6.2.4.New HPSS 6.2 System Files.....	184
6.2.5.Testing the Metadata Conversion.....	184
6.2.6.Estimating the Metadata Conversion Time (for 4.5 upgrades only).....	184
6.2.6.1.Running Time for the Long Running Metadata Conversion Utilities (for 4.5 upgrades only).....	185
6.2.7.Capturing the Metadata Conversion Output.....	185
6.2.8.DB2 Configuration and Tuning (for 4.5 upgrades only).....	186
6.2.9.Overview of the Upgrade Utilities.....	187
6.2.9.1.HPSS 4.5 and HPSS 5.1 Upgrade Utilities.....	187
6.2.9.2.HPSS 4.5 Upgrade Utilities.....	188
6.2.9.3.HPSS 5.1 Upgrade Utilities.....	189
6.3.HPSS 6.2 Upgrade Procedures.....	190
6.3.1.Verify Prerequisites.....	190
6.3.2.Acquire Software.....	190
6.3.3.Install Authentication and Authorization Mechanisms.....	191
6.3.4.Install or Upgrade DB2.....	193
6.3.5.Upgrade AIX.....	194
6.3.6.Install or Upgrade Java.....	194
6.3.7.Save Current HPSS Code and Configuration Files.....	194
6.3.8.Prepare HPSS 6.2 Code.....	194
6.3.8.1.Install HPSS 6.2 Distribution Image.....	195
6.3.8.2.Compile HPSS 6.2 Source Code (if necessary).....	195
6.3.8.3.Disable Binaries, temporarily.....	196
6.3.9.Set Environment Variables.....	196
6.3.10.Setup Authentication and Authorization.....	199
6.3.11.Pre-Conversion System Check.....	202
6.3.12.Take a full backup of SFS or DB2.....	203
6.3.13.Upgrade from HPSS 4.5 to HPSS 6.2.....	203
6.3.13.1.Prepare for the Conversion.....	203
6.3.13.2.Run <code>db_convert_collect_info</code> to Collect Metadata Information	203
6.3.13.3.Convert Configuration Metadata	204
6.3.13.4.Convert Subsystem Metadata	204
6.3.13.5.Run the Long Running Utilities.....	205
6.3.13.6.Create Core Server ACLs.....	209
6.3.13.7.Terminate the Scripting Session.....	210
6.3.13.8.Modify Permissions on Devices.....	210
6.3.14.Verify HPSS 4.5 Conversion Results	211
6.3.14.1.Capture Session Output.....	211
6.3.14.2.Run <code>db_convert_size_check</code>	211
6.3.14.3.Run <code>db_convert_ns_check</code>	212
6.3.14.4.Run <code>db_convert_address_check</code>	212
6.3.14.5.Terminate Scripting Session.....	213
6.3.15.Upgrade from HPSS 5.1 to HPSS 6.2	213

6.3.16.Enable DB2 Backup.....	215
6.3.17.Perform the DCE Export: <code>hpss_dce_export</code>	215
6.3.18.Perform the Unix, LDAP or Kerberos Import.....	215
6.3.19.Prepare the 6.2 System.....	217
6.3.19.1.Tune DB2 for normal operations.....	218
6.3.19.2.Modify Accounting, if applicable.....	218
6.3.19.3.Update FTP Configuration Files.....	218
6.3.19.4.Populate the <code>HPSS.conf</code> files.....	218
6.3.19.5.Copy the <code>rc.hpss</code> Script to <code>/etc</code>	218
6.3.19.6.Run the <code>bind</code> Script	218
6.3.19.7.Create Default Server Security ACLs.....	218
6.3.19.8.Create SSM User Ids.....	219
6.3.19.9.Create Location Server Endpoints.....	220
6.3.19.10.Perform Additional Remote Mover Configuration.....	220
6.3.20.Bring up the HPSS 6.2 Servers.....	220
6.3.20.1.Invoke the SSM System Manager, Startup Daemon and prerequisite software	221
6.3.20.2.Update HPSS Configurations.....	222
6.3.20.3.Dump Accounting Metadata, if applicable.....	223
6.3.20.4.Start HPSS 6.2 Servers.....	224
6.3.21.Verify 6.2 System.....	224
6.3.22.Clean Up After a 4.5 to 6.2 Upgrade.....	225
6.3.23.Clean Up After a 5.1 to 6.2 Upgrade.....	225
6.3.24.Revert HPSS 6.2 System to Prior Release	225
6.3.24.1.Revert the HPSS 6.2 System to Version 4.5.....	225
6.3.24.2.Revert the HPSS 6.2 System to Version 5.1.....	226
6.4.Metadata Conversion Troubleshooting Procedures.....	227
6.4.1.HPSS 4.5 to 6.2 Conversion Utility Errors and Warnings.....	227
6.4.1.1. <code>db_convert_collect_info</code> Errors.....	227
6.4.1.2. <code>db_config_convert</code> , <code>db_subsys_convert</code> , and <code>db_lr_convert</code> Errors and Warnings.....	228
6.4.2.HPSS 5.1 to 6.2 Conversion Utility Errors.....	231
6.4.2.1. <code>hpss_md_convert_51</code> Errors.....	231
6.4.2.2. <code>hpss_init_server_acls</code> Errors.....	232
6.5.HPSS 4.5 Conversion Utilities Output.....	232
6.5.1.Interpreting Output from the 4.5 Conversion Utility.....	232
6.5.2.Examples of HPSS 4.5 Conversion Utility Output.....	234
6.5.2.1. <code>db_convert_collect_info</code> Output.....	234
6.5.2.2. <code>db_config_convert</code> Output.....	234
6.5.2.3. <code>db_subsys_convert</code> Output.....	238
6.5.2.4.Long Running Conversion Utilities Output.....	241
Appendix A. Glossary of Terms and Acronyms.....	245
Appendix B. References.....	256
Appendix C. Developer Acknowledgments.....	258
Appendix D. HPSS.conf Configuration File.....	259
D.1. PFTP Client Stanza.....	259
D.2. PFTP Client Interfaces Stanza.....	264
D.3. Multinode Table Stanza.....	266
D.4. Network Option Stanza.....	268
D.5. PFTP Daemon Stanza.....	273

D.6. Transfer Agent Stanza.....	287
D.7. Stanzas Reserved for Future Use.....	291
Appendix E. hpss_env_defs.h.....	293
Appendix F. /var/hpss files.....	309

List of Figures

Figure 1. File Migration and Stage Operations.....	37
Figure 2. Class of Service / Hierarchy / Storage Class.....	38
Figure 3. HPSS Components.....	40
Figure 4. HPSS Generic Configuration.....	52
Figure 5. Basic HPSS Metadata & Filesystem Allocation.....	70
Figure 6. The Relationship of Various Server Data Structures.....	82
Figure 7. Relationship of Class of Service, Storage Hierarchy, and Storage Class.....	102

List of Tables

Table 1. HPSS Client Interface and Mover Platforms.....	49
Table 2. Supported Platform/Driver/Tape Drive Combinations.....	64
Table 3. Cartridge/Drive Affinity Table.....	66
Table 4. Paging Space Info.....	79
Table 5. Gatekeeping Call Parameters.....	100
Table 6. Suggested Block Sizes for Disk.....	107
Table 7. Suggested Block Sizes for Tape.....	108
Table 8. Network Options.....	131
Table 9. Installation Package Sizes and Disk Requirements.....	136
Table 10. Runing Times for Long Running Metadata Conversion Utilities.....	185
Table 11. 6.2 Default Server Configuration Parameters.....	222
Table 12. PFTP Client Stanza Fields.....	259
Table 13. PFTP Client Interfaces Stanza Fields.....	264
Table 14. Multinode Table Stanza Fields.....	266
Table 15. Network Options Stanza Fields.....	269
Table 16. PFTP Daemon Stanza Description.....	273
Table 17. Transfer Agent Stanza Description.....	287

Preface

About this book

The HPSS Installation Guide is for use both at system installation time as well as throughout the lifetime of the system. It will guide system administrators through the planning and installation of a new HPSS system. It also guides system administrators through the conversion process to upgrade existing HPSS systems to Release 6.2. It serves as a reference whenever the system is reconfigured by the addition, deletion, or modification of hosts, tape libraries, devices, or other components.

Chapter 1 discusses HPSS changes for Release 6.2.

Chapter 2 gives an overview of HPSS technology.

Chapters 3-5 guide administrators of new HPSS systems through planning, system preparation, HPSS software installation, and configuration of the HPSS infrastructure.

Chapter 6 guides administrators of existing 4.5 or 5.1 HPSS systems through the conversion process, bringing those systems up to Release 6.2.

Conventions Used in This Book

Example commands that should be typed at a command line will be preceded by a percent sign (%) and be presented in a boldface courier font:

```
% sample command
```

Example command output and example contents of ASCII files will be presented in a courier font:

```
sample file line 1  
sample file line 2
```

Any text preceded by a pound sign (#) should be considered comment lines:

```
# This is a comment
```

Chapter 1. Release 6.2

This chapter summarizes HPSS changes for Release 6.2 into four categories: new features, retired features, deferred features, and changed features. Changes since release 4.5 and 5.1 are described.

1.1. New Features

This section describes the new HPSS features added to Release 6.2.

1.1.1. DCE Replacement

Previous HPSS releases used DCE to provide authenticated client/server remote procedure calls. Since DCE is no longer offered as a commercial product, it has been replaced in HPSS with a solution based on commercially available authentication services and ONC RPC technology.

The new HPSS infrastructure provides four primary sub-systems: POSIX Threads, Universal Unique Identifiers (UUIDs), ONC Remote Procedure Calls and a collection of security services. The security services include support for Kerberos and an LDAP based registry service. For more information on the new HPSS infrastructure, see Section 2.3.4: *HPSS Infrastructure* on page 44.

1.1.2. Linux Support

All HPSS servers are fully supported on Linux.

1.1.3. Security

The HPSS Security implementation, which was based on DCE security, has been replaced with Unix or MIT Kerberos authentication and with Unix or LDAP authorization. Refer to Chapter 2: *Security and System Access* of the *HPSS Management Guide* and Section 3.2.7: *Security* on page 57 for more information.

1.1.4. SCSI PVR

Support for SCSI connected tape libraries has been added to HPSS 6.2. The SCSI PVR can be used to control any SCSI command based tape library.

1.1.5. HPSS VFS Interface

The HPSS VFS Interface provides a POSIX I/O interface to the HPSS filesystem. The root of the HPSS directory tree or a subdirectory can be mounted as a filesystem. HPSS files can be accessed by any software that complies with the POSIX I/O API by using the HPSS VFS Interface. For more information, please refer to Section 13.4: *HPSS VFS Interface Configuration* of the *HPSS Management Guide*.

1.1.6. GridFTP Interface

The GridFTP interface uses the PIO interface to transfer data into/from HPSS.

1.1.7. SAN3P/PIO Support

PIO can be used to issue I/O requests to direct attached HPSS SAN3P disk cache.

1.1.8. Additional hpssadm Configuration Options

- Class of Service (COS)

- Storage Class (but not subsystem-specific storage class options)
- Global configuration
- Accounting policy
- Location policy
- All server configuration. Newly supported options include Core Server, Gatekeeper, Location Server, Log Daemon, Migration/Purge Server, PVL, all PVRs, and SSM. The Mover, Log Client, and Startup Daemon were previously supported and still are.
- Modifying devices and drives. This is the equivalent functionality of modifying fields on the device info and drive info screens in the GUI.

1.1.9. Additional hpssadm operations

- Import/export
- Create/delete resources
- Task monitoring. A new "task" command displays the status of any long running background tasks submitted during the session, such as imports or resource creates.
- Specifying volumes from a file for import/export/delete/move. hpssadm, like the GUI, now allows the user to submit a file listing the target volumes for import, export, delete resources, and cartridge move operations. This is an alternative to the previous capability of building the list on the screen with the Fill Count, Fill Increment, and Volume Label fields, which is still provided. As a result of adding this capability, the group labels on the structures have changed so scripts which use the old method to specify the volume list will need a slight modification. See the hpssadm man page for examples.

1.1.10. Additional Library and Device Support

- Spectra Logic library
- ADIC Scalar i500 library
- IBM and HP LTO Generation 3 and 4 drives. Tape encryption has not been certified with HPSS.
- SAIT-1 drives
- IBM TS1120 (3592 Generation 2 and 3) drives. Tape encryption has not been certified with HPSS.
- Sun StorageTek T10000 drives
- Sun StorageTek 9840D drives
- DataDirect Networks (DDN) disks

1.1.11. SAN Virtual Volume ID Mapping

A feature to consistently map device pathnames to volumes. A unique ID is assigned to each volume. The IDs are then mapped to device pathnames.

1.1.12. Drive Pools

HPSS provides HPSS end clients the ability to direct tape read I/O requests to a predefined group of tape drives referred to as a Drive Pool. This ability helps HPSS administrators manage tape drive scheduling and thus availability. For more information, please refer to Section 7.3: *Drive Pools* of the *HPSS Management Guide*.

1.1.13. FTP Enhancement

- FTP application mode: FTP can be compiled as either a 32-bit or a 64-bit application, by setting the BIT64_SUPPORT flag on or off in the Makefile.macros file, when building with the HPSS FTP source extracted with "build-ftp" option. The 64-bit Kerberos libraries will be required to support Kerberos authentication.
- PFTP client is now supported on Solaris 10 (Intel/AMD only).

1.1.14. mkhpss Enhancement

A new option is provided by mkhpss to configure the DB2 log mirroring capability to better protect the DB2 transaction log files. DB2 log mirroring allows the database to write an identical second copy of log files to a different path. The DB2 log files are essential in protecting a database and enable transactions occurring after a database backup to be recovered. Loss of the DB2 log files can result in significant impact to the operation of the system, extensive downtime, and potential loss of data. It is vital that DB2 log files be stored on highly reliable disk systems. In addition, DB2 log mirroring must be used to protect the DB2 log files on two separate disk systems. The disk systems must also be protected using RAID1 or RAID5 configurations. This mkhpss enhancement can only be used for new HPSS installations. For existing sites, please consult with your HPSS Support Representative for assistance.

1.1.15. DB2 Monitoring

A new feature to have the HPSS DB2 Log Monitor periodically checks the DB2 transaction log files to make sure that the primary and mirror logs are congruent and performing normally. It also scans the DB2 diagnostic log for indications of problem with the DB2 log files. For each identified problem, it will issue an HPSS alarm and indicate the path of the diagnostic log and the line number where the error is reported. The administrator should then open the DB2 diagnostics file and examine the reported problem in detail.

1.1.16. File Family enhancements

The number of allowed file families is increased to 4,294,967,295. In addition, authorized callers can assign the File Family ID of a file via the hpss_FileSetAttributes Client API function, independently of the underlying fileset.

1.1.17. SSM Configuration Files Consolidation

The SSM configuration files ssm_krb5.conf and ssm_unix.conf have been consolidated into a single configuration file called ssm.conf. Users may continue using their existing configuration file by specifying the "-m" option to the hpssadm.pl or hpssgui.pl script, or they may rename their existing configuration file to ssm.conf. For environments with multiple authentication mechanisms, it may be desirable to maintain multiple configuration files.

1.1.18. Mover Enhancement

Multiple Movers for one or more HPSS instances can now be configured to run on the same machine. The `-c <alternate var path>` flag is added to the Mover entry in the `inetd` configuration file to specify an alternate `"/var/hpss"` path to be used by the Mover. In addition, the `-s` flag is added to enable syslog logging for the Mover. This feature is intended for multiple HPSS test systems to share the same set of Mover machines.

1.2. Retired Features

- HPSS is no longer supported fully on Solaris. HPSS Mover and HPSS clients are still being supported on Solaris.
- The **settapestats** utility program is no longer supported. The 6.2 Core Server calculates tape volume statistics when it starts.
- STK RAIT PVR is no longer supported.
- LTO PVR is no longer supported on Linux. The SCSI PVR should be used for LTO drives on Linux.
- Non-DCE Gateway is no longer supported.
- Federated Name Space and Remote Site Policy are not supported in 6.2.
- Mirrored Filesets are no longer supported.
- The terminology of "Non-DCE" Movers is no longer being used. It is now referred to as the Mover.
- The terminology of "Non-DCE" Client API is no longer being used. It is now referred to as the Client API.
- The `hpss_PVrr` utility is no longer supported. The source code for this utility can be found in `/opt/hpss/tools/unsupported`.
- The `hpss.clean` command is no longer supported.
- The HPSS Metadata Backup Tools are no longer supported.

1.3. Deferred Features

XFS is not supported in HPSS 6.2. XFS references have been left in the HPSS documentation to support the option of re-enabling XFS in future releases.

1.4. HPSS Changes

This section describes the changes made to HPSS 6.2.

1.4.1. Documentation Organization Changes

The SSM Guide is now merged with the Management Guide.

1.4.2. Metadata Changes

- Added the AUTHZACL table to maintain security ACLs for HPSS servers and for SSM client access to SSM.
- Deleted the DISKSPACE table. The disk space allocation maps are now maintained in the

Core Server's memory image

- Modified the DMG table. In support of the new HPSS RPC library, the TCP port was eliminated and the Program and Version numbers were added to DMG specific configuration.
- Modified the DMGFILESET table. The TCP port was eliminated and the TCP hostname and RPC endpoint information was added.
- Modified the GATEKEEPER table. The length of site policy pathname was changed from 127 to 1023 characters.
- Modified the LSPOLICY table. The group name was eliminated and realm name was added. Changed length of local site name from 31 to 255 characters.
- Modified the MOVERDEVICE table. In support of the new SAN3P capabilities, SanID was added to mover device configuration. Since this is new to HPSS 6.2, these will be set to 0 for all mover devices.
- The NDCG table is now obsolete. It has been renamed, but left intact.
- Modified the NFS table. In support of the new authentication mechanisms, the credential object ID has been eliminated. Privileged caller principal length was changed from 15 to 255 characters.
- SSSTATS table is now obsolete. It has been renamed, but left intact.
- Modified the SERVER table. In support of the new HPSS RPC library, the authorization service and authentication service information has been eliminated. Request queue size information, RPC program, version number, and authentication mechanism information has been added, along with new index definitions.
- Added the SERVERINTERFACES table. This table is populated by the metadata converter programs with newly created server interface information.
- Modified the SITE table. The descriptive and LS group names have been eliminated, and site and realm names were added. The pre-6.2 SITE table will be renamed to PRE62_SITE but the metadata in the table will not be converted into the new SITE table since remote sites are no longer supported.
- Modified the STORAGESEGDISK table. A new index is required for this table to support the new disk space allocation mechanism.

1.4.3. DMAP Gateway Changes

Since DFS is no longer supported, the logic to support DFS in the DMAP Gateway has been removed. DMAP Gateway will only support XFS in 6.2. Changes were also made in the underlying RPC mechanism used for communication with the HDM. The ability to delete the DMAP Gateway and Core Server components of an XDSM Fileset have been removed. Changes made to the SSM windows to support XFS are described below.



Information

1.4.3.1. Creating an XDSM Fileset

- The “Create HPSS/XDSM Fileset” window has been renamed ”Create XDSM Fileset”. This window is currently not accessible since XFS is not supported.
- Fields specific to DFS fileset creation have been removed. Since HPSS managed XFS

filesystems do not support mirrored namespaces, fields which were specific to managing mirrored filesets are also no longer available. This includes the following configuration options:

- Global Mount Point
- Local Mount Point
- Fileset Owner
- Fileset Permissions
- Communication between the HDM and the DMAP Gateway now uses the HPSS RPC library. Fields which were used to specify TCP communication information have been removed.

1.4.3.2. Viewing DMAP Gateway XDSM Fileset Information

- Fields and functions which were needed for DFS or mirrored fileset management have been removed. This includes the following changes:
 - The ability to change the name of an XDSM Fileset.
 - The ability to change the fileset state (i.e.: mounted, pre-unmounted etc.).
 - The "Name Server Object Handle of Fileset Mountpoint" data is no longer shown.
 - The ability to delete the DMAP Gateway portion of an XDSM Fileset independently from the core server is no longer supported.
- Fileset information and associated statistics are now shown on a single window. The "Requests via TRPC (HPSS) Interface" statistics were specific to mirrored filesets and have been removed.
- The RPC endpoint for the HDM is shown instead of the "HPSS/DMAP TCP Hostname" and "HPSS/DMAP TCP Port" fields. This reflects the change in the underlying communication mechanism used between the HDM and the DMAP Gateway. Additionally, these fields may no longer be modified from the DMAP Gateway Fileset Information window.

1.4.3.3. Viewing Core Server XDSM Fileset Information

- The name of this window has been changed from "Name Server Fileset Information" to "Core Server Fileset Information".
- The ability to modify the following fields from this window is no longer allowed:
 - Fileset ID
 - Fileset Name
 - User and Group ID
 - Fileset Type
 - Permissions
- The following informational fields have been removed:
 - Name Server Object Handle

1.4.4. SSM Changes

Significant changes were made in SSM between Releases 4.5 and 5.1 and again between 5.1 and 6.2. For the reader's convenience, all changes between 4.5 and 6.2 are summarized in Section 3.3.4.1: *Changes Affecting Sites Upgrading Directly from 4.5*. Changes between 5.1 and 6.2 are summarized in Section 3.3.4.2: *Changes Affecting Sites Upgrading from 5.1*. The tables indicate whether each change affects the SSM Server (the System Manager), the SSM GUI hpssgui, and/or the HPSS command line utility hpssadm. Sites need consult only the table which pertains to them.

1.4.4.1. Changes Affecting Sites Upgrading Directly from 4.5

<i>Changes since 4.5</i>	<i>Server</i>	<i>GUI</i>	<i>ADM</i>
The SSM System Manager and Data Server were consolidated in release 6.2 into a single server, the SSM System Manager.	●		
Java is now required for the SSM Graphical User Interface (hpssgui). Java is no longer required by the SSM server (the new consolidated SSM System Manager).	●	●	
SSL is no longer supported with SSM. This means it is no longer necessary to create an SSL certificate (the /var/hpss/ssm/ds.cer file) or import it into the certificate authority file (\$JAVA_ROOT/jre/lib/cacerts) on each SSM client machine.	●	●	●
The Java Security Manager is no longer used with SSM. This means the Java Policy files are no longer required. In 4.5, these files were the java.policy.ds and the java.policy.hpssadm.	●	●	●
The hpssadm.jar and mobjects.jar files are no longer used. All classes are now in a single jar file hpss.jar.		●	●
The start_ssm script is no longer available. The SSM System Manager is started with the rc.hpss script. The rc.hpss script has been moved from a default location of /etc to /opt/hpss/bin.	●		
The start_ssm_session script is no longer available. In HPSS 6.2, the SSM GUI is started with the hpssgui startup script. There are perl and vbs versions of this script, named with an extension to indicate the type of script: hpssgui.pl and hpssgui.vbs.		●	
There are new versions of the hpssadm startup scripts in perl and vbs. These are named with an extension to indicate the type of script: hpssadm.pl and hpssadm.vbs. The ksh and bat scripts are no longer supported.			●
Options to the new hpssgui startup scripts differ considerably from the options to the old start_ssm_session script. Options to the hpssadm startup scripts have also changed significantly. See the man pages for details. Following are some highlights:		●	●
The scripts are dependent upon an SSM configuration file (ssm.conf), created by mkhpss, which contains some site-specific configuration values. The SSM client script will attempt to ready the SSM configuration file in the current working directory unless the pathname to this file is specified by the -m option.		●	●

<i>Changes since 4.5</i>	<i>Server</i>	<i>GUI</i>	<i>ADM</i>
<p>The SSM client scripts now use an internal polling mechanism for getting window updates (as opposed to being notified of the update by the server). This means that:</p> <ul style="list-style-type: none"> • The SSM client application no longer requires a second port for two-way communication, • The rate of polling can be fine-tuned by the user (see the -A, -G, -L -M, -W options). 		●	●
<p>The scripts are dependent upon a login.conf file and, if using Kerberos authentication, a krb5.conf file. Both files are created by hpssuser. The scripts will look in the current working directory for these files unless they are specified on the command line (-C and -k options).</p>		●	●
<p>It is no longer necessary to add a security provider entry to the java system java.security file.</p>		●	●
<p>The pathname to the java executable can be specified by using the -j option. If this option isn't specified, then the script will use the java executable in \$JAVA_BIN if it is valid. If \$JAVA_BIN/java is not defined, then the script will use 'java' which will use the client's \$PATH.</p>		●	●
<p>Effective with HPSS 5.1.1, the hpssgui script supports customizing the user's background color or Look & Feel using the -b, -F and -T options.</p>		●	
<p>The SSM user can specify the number of alarms to display on the alarm list using the -N option.</p>		●	●
<p>The SSM user can specify the pathname to the hpss.jar using the -P option.</p>		●	●
<p>The SSM user can specify the path to search for application preferences using the -i option. This path is now local to the SSM client machine; preferences are no longer stored on the SSM server machine.</p>		●	●
<p>Sites that are converting from a previous release of HPSS should replace all their hpssgui and hpssadm startup scripts on all client platforms with the new versions.</p>		●	●
<p>In HPSS 6.2, ssh tunneling communications between the SSM client and server is supported.</p>	●	●	●
<p>The SSM clients can contact the System Manager across a Virtual Private Network connection (VPN). See the -p and -h options on the hpssgui and hpssadm man pages.</p>	●	●	●

<i>Changes since 4.5</i>	<i>Server</i>	<i>GUI</i>	<i>ADM</i>
<p>The SSM client scripts can use ports exempted by the network administrator as firewall exceptions. See the -n option on the hpssadm/hpssgui man pages.</p> <p>The port on which the System Manager will listen may be controlled by setting the \$HPSS_SSM_SERVER_LISTEN_PORT environment variable. The default setting is 0 (zero) which means that the port will be chosen by the RPC portmapper. See Section 3.3.6: <i>Using SSM Through a Firewall, of the HPSS Management Guide.</i></p>	●	●	●
<p>Access to port 88 is needed for Kerberos authentication. Access to port 111 is needed if the RPC portmapper is needed to find the System Manager.</p>	●	●	●
<p>The user_authorization.dat and hpssadm.config files provided in HPSS 4.5 are no longer available.</p> <p>In 6.2, all authorized SSM users and their privilege levels are defined in the DB2 user acl AUTHZACL table. See Section 3.3.2.1: <i>hpssuser Utility</i> and Section 3.3.2.2: <i>SSM User Authorization</i> in the <i>HPSS Management Guide</i> for details.</p>	●	●	●
<p>The hpssgui and hpssadm programs will now optionally record every error and informational message in an ASCII log file. The "-S" option for both programs is used to specify whether to create the session log and what to name it.</p>		●	●
<p>The basic and specific server config structures, along with the server's log policy, have been combined in 5.1 into a single structure.</p>		●	●
<p>SSM now represents the three basic Core Server volume structures, the physical volume, virtual volume, and storage map, as a single combined Core Server volume structure.</p>		●	●
<p>The delog utility is no longer available via the SSM GUI. This feature has been retired.</p>		●	
<p>The ability to broadcast messages to other SSM GUI users is not yet available.</p>		●	
<p>Alarms and events are no longer acknowledged from the SSM GUI.</p>		●	
<p>The SSM clients can specify the number of items to display for the Alarms and Events list. See the -N option on the hpssadm/hpssgui man page and Section 9.6: <i>Managing SSM Alarms and Events</i> in the <i>HPSS Management Guide.</i></p>		●	●
<p>The SSM clients keep an internal cached copy of the Alarms and Events list which is maintained by regular polling requests to the System Manager. Each polling request returns only "new" messages, those which have not already been retrieved by the SSM client in a previous polling request. If there are no new messages in the System Manager log cache, no messages are returned. This will help performance of the display of this window. See the -A, -G and -N options of the hpssadm/hpssgui man pages and Section 9.6.5: <i>Controlling SSM Log Message handling</i> in the <i>HPSS Management Guide.</i></p>		●	●

<i>Changes since 4.5</i>	<i>Server</i>	<i>GUI</i>	<i>ADM</i>
The ability to see which users are logged into SSM, referred to in HPSS 4.5 as a list of SSM "consoles", is available in 6.2 as part of the System Manager Statistics window from both the hpssgui and hpssadm.		●	●
The menu bar has been reorganized extensively. The "Set Keyboard", "Show Sammi Environment", "View Sammi Errlog", and "About Sammi" items from the 4.5 Session menu and the "SSM consoles" item from the 4.5 Monitor menu are no longer available. Some of the information about the gui session and data server which was formerly available from these screens is now available in 6.2 from the "System Manager Statistics" and "User Session Information" items from the "SSM Information" submenu of the "Monitor" menu. Some information is also available in the new user session log. See Chapter 3: <i>Using SSM</i> in the <i>HPSS Management Guide</i> for a full description of the layout of the menu and a description of the functions available from each menu item.		●	
The Monitor, Operations, and Configure menus are available in a menu tree on the bottom half of the Health and Status window. The tree is organized exactly as the corresponding items from the menu bar, and the functionality of each item is the same as its corresponding item on the menu bar.		●	
The View menu of the Health and Status screen allows some portions of the screen to be hidden or redisplayed.		●	
The PVR Cartridge Threshold is included in the HPSS Status section of the Health and Status information. When the status is something other than OK , the <i>PVR Information</i> button on the GUI screen will become activated; clicking this button will bring up a window for each PVR cartridge that is not healthy.		●	●
To support users with color blindness or other visual problems, all text is now displayed on white or light gray backgrounds. Color (green, yellow, red) is still used to indicate the severity level of alarms and component statuses, but the color is confined to a small sphere displayed next to the text. For example, a major alarm might be displayed on the Alarms and Events screen as: ● May 2, 2005 11:26:13 AM Major PVL Bad things happened		●	
More than one preference may now be defined for each list. See the description of the "preferences combo box" in Section 3.6: <i>Common Window Elements</i> of the <i>HPSS Management Guide</i> for details.		●	
Note that the mechanism for selecting columns to be displayed in each list window has moved from the list's Preferences window to the "Column View" menu of the list window itself. See the description of the "Column View" menu item in Section 3.6: <i>Common Window Elements</i> of the <i>HPSS Management Guide</i> for details.		●	

<i>Changes since 4.5</i>	<i>Server</i>	<i>GUI</i>	<i>ADM</i>
Column ordering is now controlled by dragging columns to the desired location. The modified order is preserved automatically in the user preferences across hpssgui restarts.		●	
When messages have been written to the status bar, the most recent messages can be viewed in the status bar's tooltip. Rolling the mouse over the status bar without clicking gives a tooltip that says, "Click mouse in status bar to view messages" if there are status messages to view. If there are no status messages then the tooltip says, "No status messages". This message stays up for about 4 seconds or until the user moves the mouse out of the status bar area. To view up to the last STATUS_MSG_MAX (30) messages that have been written to the status bar, click on the status bar. The tooltip that is displayed will show up to the last 30 messages and will remain visible until the mouse is moved out of the status bar or for 10 minutes.		●	
Most user updates to fields on information windows are no longer sent to the server immediately or automatically. The Update button on the window must be clicked to submit the update to the server. Fields modified by the user will be flagged by a diskette icon to indicate the window copy has been modified but not saved.		●	
The list of volumes for import, export, resource deletes, and cartridge move operations may now be specified from an external input file as an alternative to building the list on the hpssgui screen or within the hpssadm operation.		●	●
<p>Since the basic and specific server configuration structures have been combined into a single structure, there is no longer a need for a -server_type option to the hpssadm config command. This option is no longer used.</p> <p>The types of configuration structures supported by the -type option now are specified by the title names used on the config struct windows. See the hpssadm man page for a list of supported config types.</p> <p>Any type name may be abbreviated so long as the abbreviation is unique. See the hpssadm man page for details.</p>			●

<i>Changes since 4.5</i>	<i>Server</i>	<i>GUI</i>	<i>ADM</i>
<p>The HPSS 4.5 hpssadm commands</p> <ul style="list-style-type: none"> • pvr_cartridge • pvl_volume • ss_pv • ss_map • ss_vv <p>have been replaced by a single command "volume". The volume command has a required option "-type" for which these types may be specified:</p> <ul style="list-style-type: none"> • PVL Volume Information • PVR Cartridge Information • CS Disk Volume • CS Tape Volume <p>The new volume command behaves similarly to the config command, where the types are the titles of the window screens for the structures. The type names may be abbreviated so long as the abbreviation is unique.</p> <p>The volume command has several subcommands which support displaying and updating the volume, importing and exporting volumes, creating and deleting resources, and moving cartridges.</p> <p>There is no longer a separate command for locking or unlocking a volume in 6.2. This has been replaced by the practice of setting the VV Condition and Retired fields structure on the storage map in the disk or tape volume structure. (This is a core server design change.) The fields of the map structure may be modified with the update command. The update command may also be used on any of the volume commands to modify any settable field.</p>			●
<p>A new hpssadm command "ssm" is supplied in 6.2 for shutting down the System Manager and for displaying the status of the System Manager and of the hpssadm session.</p>			●
<p>All available columns of each SSM list (server, device, storage class, pvl job, alarm) are now displayed by the hpssadm list commands in the system default order. At this time hpssadm has no facility for customizing the list format as the gui does.</p>			●
<p>Field layout and grouping for each structure displayed by the hpssadm now matches the order and grouping used by the corresponding window of the SSM GUI.</p>			●
<p>Most hpss structures have changed so hpssadm scripts used before 5.1 may need to be modified to use the new field and subfield names.</p>			●

<i>Changes since 4.5</i>	<i>Server</i>	<i>GUI</i>	<i>ADM</i>
The hpssadm config command now supports the additional structures: <ul style="list-style-type: none"> • Class of Service Config • Storage Class (but not subsystem-specific storage class options) • Global Config • Accounting Policy • Location Policy • All server configurations; servers which were not supported before are the core server, gatekeeper, location server, log daemon, migration/purge server, pvl, all pvrs, and ssm. The mover, log client and startup daemon are still supported as previously. 			●
A new command “task” displays the status of any long running background tasks submitted during the session, such as imports or resource creates.			●
A new subcommand “update” has been added to the hpssadm device command for updating the mover device and pvl drive objects.			●

1.4.4.2. Changes Affecting Sites Upgrading from 5.1

<i>Changes since 5.1</i>	<i>Server</i>	<i>GUI</i>	<i>ADM</i>
The SSM System Manager and Data Server were consolidated in release 6.2 into a single server, the SSM System Manager.	●		
Java is no longer used in the server side of SSM (the new consolidated SSM System Manager).	●		
SSL is no longer supported with SSM. This means it is no longer necessary to create an SSL certificate (the /var/hpss/ssm/ds.cer file) or import it into the certificate authority file (\$JAVA_ROOT/jre/lib/cacerts) on each ssm client machine.	●	●	●
The Java Security Manager is no longer used with SSM. This means the Java policy files are no longer required. In 5.1, these files were the java.policy.ds and the java.policy.ssmuser.	●	●	●
The start_ssm script is no longer available. The SSM System Manager is started with the rc.hpss script. The rc.hpss script has been moved from a default location of /etc to /opt/hpss/bin.	●		
There are new versions of the hpssgui and hpssadm startup scripts in perl and vbs. These are named with an extension to indicate the type of script: hpssadm.pl, hpssadm.vbs, hpssgui.pl and hpssgui.vbs. The ksh and bat scripts are no longer supported.		●	●
The scripts are no longer distributed as templates; they will be packaged into the \$HPSS_PATH_BIN (default /opt/hpss/bin) directory.			

<i>Changes since 5.1</i>	<i>Server</i>	<i>GUI</i>	<i>ADM</i>
Options to the hpssgui and hpssadm startup scripts have changed significantly. See the man pages for details. Following are some highlights:		●	●
The scripts are dependent upon an ssm configuration file (ssm.conf), created by mkhpss, which contains some site-specific configuration values. The SSM client script will attempt to read the ssm configuration file in the current working directory unless the pathname to this file is specified by the -m option.		●	●
The SSM client scripts now use an internal polling mechanism for getting window updates (as opposed to being notified of the update by the server). This means that: <ul style="list-style-type: none"> • the SSM client application no longer requires a second port for two-way communication, • using SSM across an ssh tunnel is simpler since you now need to set up the tunnel only in one direction, • the rate of polling can be fine-tuned by the user (see the -A, -G, -L -M, -W options). 		●	●
The scripts are dependent upon a login.conf file and, if using Kerberos authentication, a krb5.conf file. The hpssuser program creates both of these files. The scripts will look in the current working directory for these files unless they are specified on the command line (-C and -k options).		●	●
It is no longer necessary to add a security provider entry to the java system java.security file.		●	●
The pathname to the java executable can be specified by using the -j option. If this option isn't specified, then the script will use the java executable in \$JAVA_BIN if it is valid. If \$JAVA_BIN/java is not defined, then the script will use 'java' which will use the client's \$PATH.		●	●
Effective with HPSS 5.1.1, the hpssgui script supports customizing the user's background color or Look & Feel using the -b, -F and -T options. Sites which did not apply the 5.1 patch would not have had this functionality.		●	
The SSM user can specify the number of alarms to display on the alarm list using the -N option.		●	●
The SSM user can specify the pathname to the hpss.jar using the -P option.		●	●
The SSM user can specify the path to search for application preferences using the -i option. This path is now local to the SSM client machine; preferences are no longer stored on the SSM server machine.		●	●
Sites which are converting from a previous release of HPSS should be certain to replace all their hpssgui and hpssadm startup scripts on all client platforms with the new versions.		●	●

<i>Changes since 5.1</i>	<i>Server</i>	<i>GUI</i>	<i>ADM</i>
The SSM client script options for connecting to the System Manager across a Virtual Private Network connection (VPN) have changed. See the -p and -h options on the hpssgui and hpssadm man pages.	●	●	●
The SSM client script option for using ports exempted by the network administrator as firewall exceptions has changed; see the -n option on the hpssadm/hpssgui man pages. The port on which the System Manager will listen may be controlled by setting the \$HPSS_SSM_SERVER_LISTEN_PORT environment variable. The default setting is 0 (zero) which means that the port will be chosen by the RPC portmapper. See Section 3.3.6: <i>Using SSM Through a Firewall, of the HPSS Management Guide.</i>	●	●	●
Access to port 88 is needed for Kerberos authentication and access to port 111 is needed if the RPC portmapper is needed to find the System Manager.	●	●	●
The ssmuser.config file provided in HPSS 5.1 is no longer available. In 6.2, all authorized SSM users and their privilege levels are defined in the DB2 user acl AUTHZACL table. See Section 3.3.2.1: <i>hpssuser Utility</i> and Section 3.3.2.2: <i>SSM User Authorization</i> in the <i>HPSS Management Guide</i> for details.	●	●	●
SSM now represents the three basic Core Server volume structures, the physical volume, virtual volume, and storage map, as a single combined Core Server volume structure.		●	●
Access to the delog utility via the SSM GUI, deferred in 5.1, has now been permanently retired.		●	
The ability to broadcast messages to other SSM GUI users is not yet available.		●	
The SSM clients can specify the number of items to display for the Alarms and Events list. See the -N option on the hpssadm/hpssgui man page and Section 9.6: <i>Managing SSM Alarms and Events</i> in the <i>HPSS Management Guide</i> .		●	●
The SSM clients keep an internal cached copy of the Alarms and Events list which is maintained by regular polling requests to the System Manager. Each polling request returns only “new” messages, those which have not already been retrieved by the SSM client in a previous polling request. If there are no new messages in the System Manager log cache, no messages are returned. This will help performance of the display of this window. See the -A, -G and -N options of the hpssadm/hpssgui man pages and Section 9.6.5: <i>Controlling SSM Log Message Handling</i> in the <i>HPSS Management Guide</i> .		●	●
The ability to see which users are logged into SSM, referred to in HPSS 4.5 as a list of SSM "consoles", was not yet available in 5.1. It is now available in 6.2 as part of the System Manager Statistics window from both the hpssgui and hpssadm.		●	●

<i>Changes since 5.1</i>	<i>Server</i>	<i>GUI</i>	<i>ADM</i>
<p>The menu bar has been reorganized slightly.</p> <p>The "Data Server Statistics" menu item has been replaced by the "System Manager Statistics" menu item, available from the Monitor->SSM Information menu path.</p> <p>"Column View" was added to the menu bar for SSM windows that display an SSM table. See Section 3.6: <i>Common Window Elements</i> of the <i>HPSS Management Guide</i>. In HPSS 5.1, this ability was part of the Preference window associated with the SSM table window.</p> <p>The "User Session Log" is no longer available from the SSM GUI; the user can still view this information by using the -S option of the SSM client script and viewing the ASCII text of the session log. Since this was removed from the menu bar, the Monitor->Logging menu path which contained two sub-items (Log Files Info and User Session Log) was replaced by Monitor->Log Files Information menu path.</p> <p>The Monitor->Lookup HPSS Objects->Security menu path option has been removed.</p> <p>The Operations->Filesets & Junctions->Create XDSM Fileset menu path option was added. However, this option is currently hidden since XFS is not supported.</p> <p>Since the two SSM Servers (Data Server and System Manager) were merged into the single System Manager, all references to the Data Server in the Operations->Shutdown menu were removed.</p> <p>Configure->Restricted Users menu path option was added. It currently supports listing the restricted users and a "Reload List" button which causes the Core Servers to reread the HPSS_RESTRICTED_USER_FILE. See Chapter 2: <i>Security and System Access</i> in the <i>HPSS Management Guide</i> for a full description of restricting user access to HPSS.</p> <p>Since the SSM Guide was merged into the HPSS Management Guide and HPSS Installation Guide, the Help->HPSS Books->SSM Guide menu path option was removed.</p> <p>See Chapter 3: <i>Using SSM</i> in the <i>HPSS Management Guide</i> for a full description of the layout of the menu and a description of the functions available from each menu item.</p>		●	
<p>The PVR Cartridge Threshold is included in the HPSS Status section of the Health and Status information. When the status is something other than OK, the <i>PVR Information</i> button on the GUI screen will become activated; clicking this button will bring up a window for each PVR cartridge that is not healthy.</p>		●	●
<p>The "Sick" preference may no longer be edited.</p>		●	

<i>Changes since 5.1</i>	<i>Server</i>	<i>GUI</i>	<i>ADM</i>
The mechanism for selecting columns to be displayed in each list window has moved from the list's Preferences window to the "Column View" menu of the list window itself. See the description of the "Column View" menu item in Section 3.6: <i>Common Window Elements</i> of the <i>HPSS Management Guide</i> for details.		●	
Column ordering is now controlled solely by dragging columns to the desired location. The modified order is now preserved automatically in the user preferences across hpssgui restarts.		●	
List tables have a field that shows the number of displayed and total items in the list in the format X/Y where X is the number of items displayed and Y is the total number of items in the list. The field is left justified under the table. The X and Y values will differ if preferences are set to filter some items out of the list.		●	
The button panel to the right of the list can be hidden or displayed by clicking the tall, thin button between the list and button panel labeled ' '. If the button is pressed when the panel is displayed, the button panel will hide, allowing more space for the list. The button panel may be re-displayed by pressing the ' ' button again.		●	
Sites which are converting from HPSS 5.1 should remove all old user preference files. New preference files will be created automatically for the user when he first logs in. The user will be required to reset any preference settings (i.e. preferences will not be converted).		●	
The System Manager connection indicator is not displayed on every window; it is only displayed on the <i>Health and Status</i> Window.		●	
When messages have been written to the status bar, the most recent messages can be viewed in the status bar's tooltip. Rolling the mouse over the status bar without clicking gives a tooltip that says, "Click mouse in status bar to view messages" if there are status messages to view. If there are no status messages then the tooltip says, "No status messages". This message stays up for about 4 seconds or until the user moves the mouse out of the status bar area. To view up to the last 30 (STATUS_MSG_MAX) messages that have been written to the status bar, click on the status bar. The tooltip that is displayed will show up to the last 30 messages and will remain visible until the mouse is moved out of the status bar or for 10 minutes.		●	
The list of volumes for import, export, resource deletes, and cartridge move operations may now be specified from an external input file as an alternative to building the list on the hpssgui screen or within the hpssadm operation.		●	●
Most hpss structures have changed so hpssadm scripts used before 5.1 may need to be modified to use the new field and subfield names.			●

<i>Changes since 5.1</i>	<i>Server</i>	<i>GUI</i>	<i>ADM</i>
<p>Since SSM now represents the three Core Server volume structures as a single structure, the types specified to the hpssadm volume command have changed. The 5.1 types:</p> <ul style="list-style-type: none"> • Disk Storage Map Information • Disk Physical Volume Information • Disk Virtual Volume Information <p>have been replaced in 6.2 by the single type:</p> <ul style="list-style-type: none"> • CS Disk Volume 			●
<p>Similarly, the 5.1 types:</p> <ul style="list-style-type: none"> • Tape Storage Map Information • Tape Physical Volume Information • Tape Virtual Volume Information <p>have been replaced in 6.2 by the single type:</p> <ul style="list-style-type: none"> • CS Tape Volume 			●
<p>The hpssadm volume command now supports volume import and export, resource creation and deletion, and cartridge moves.</p>			●
<p>On the Active Storage Classes window, the “Force Migrate” has been replaced with a “Migration Controls” drop-down allowing the Administrator or Operator to perform migration operations on all the selected storage classes. Likewise, the “Force Purge” has been replaced with a “Purge Controls” drop-down allowing the Administrator to perform purge operations on all the selected storage classes.</p>		●	
<p>The "-ds" option is no longer available for the ssm "info" or "shutdown" commands. A new option "-sm" is available for the "info" command.</p>			●
<p>The hpssadm config command now supports the additional structures:</p> <ul style="list-style-type: none"> • Class of Service Config • Storage Class (but not subsystem-specific storage class options) • Global Config • Accounting Policy • Location Policy • All server configurations; servers which were not supported before are the core server, gatekeeper, location server, log daemon, migration/purge server, pvl, all pvrs, and ssm. The mover, log client and startup daemon are still supported as previously. 			●
<p>A new hpssadm command “task” displays the status of any long running background tasks submitted during the session, such as imports or resource creates.</p>			●

<i>Changes since 5.1</i>	<i>Server</i>	<i>GUI</i>	<i>ADM</i>
A new subcommand “update” has been added to the hpssadm device command for updating the mover device and pvl drive objects.			●

Chapter 2. HPSS Basics

2.1. Introduction

The High Performance Storage System (HPSS) provides hierarchical storage management and services for very large storage environments. HPSS may be of interest to organizations having present and future scalability requirements that are very demanding in terms of total storage capacity, file sizes, data rates, number of objects stored, and numbers of users. HPSS is part of an open, distributed environment based on remote procedure calls, Kerberos, LDAP directory systems and DB2 which form the infrastructure of HPSS. HPSS is the result of a collaborative effort by leading US Government supercomputer laboratories and industry to address very real, urgent high-end storage requirements. HPSS is offered commercially by IBM.

HPSS provides scalable parallel storage systems for highly parallel computers as well as traditional supercomputers and workstation clusters. Concentrating on meeting the high end of storage system and data management requirements, HPSS is scalable and designed to store up to petabytes (10^{15}) of data and to use network-connected storage devices to transfer data at rates up to multiple gigabytes (10^9) per second.

HPSS provides a large degree of control for the customer to manage the hierarchical storage system. Using configuration information defined by the customer, HPSS organizes storage devices into multiple storage hierarchies. Based on policy information defined by the customer and actual usage information, data are then moved to the appropriate storage hierarchy and to appropriate levels in the storage hierarchy.

2.2. HPSS Capabilities

A primary goal of HPSS is to move large files between storage devices and parallel or clustered computers at speeds many times faster than today's commercial storage system software products and to do this in a way that is more reliable and manageable than is possible with current systems. In order to accomplish this goal, HPSS is designed and implemented based on the concepts described in the following subsections.

2.2.1. Network-centered Architecture

The focus of HPSS is the network, not a single processor as in conventional storage systems. HPSS provides servers that can be distributed across a high performance network to provide scalability and parallelism. The basis for this architecture is the IEEE Mass Storage System Reference Model, Version 5.

2.2.2. High Data Transfer Rate

HPSS achieves high data transfer rates by eliminating overhead normally associated with data transfer operations. In general, HPSS servers establish and control transfer sessions but are not involved in actual transfer of data.

2.2.3. Parallel Operation

The HPSS Client Application Program Interface (Client API) supports parallel or sequential access to storage devices by clients executing parallel or sequential applications. HPSS also provides a Parallel File Transfer Protocol. HPSS can even manage data transfers in a situation where the number of data sources differs from the number of destination sources. Parallel data transfer is vital in situations that demand fast access to very large files.

2.2.4. Based on Standard Components

HPSS runs on UNIX and is written in ANSI C and Java. It uses remote procedure calls, a selectable security service (Kerberos or UNIX), UNIX or LDAP for user configuration information, and DB2 as the basis for its portable, distributed, transaction-based architecture. These components are offered on many vendors' platforms.

The full HPSS system has been implemented on IBM AIX and LINUX platforms, and some components of HPSS have been ported to other platforms. Refer to Section 2.4: *HPSS Hardware Platforms* on page 48 and Section 3.3: *Prerequisite Software Considerations* on page 58 for specific information.

To aid vendors and users in porting HPSS to new platforms, HPSS source code is available.

2.2.5. Data Integrity Through Transaction Management

Transactional metadata management, provided by DB2, enables a reliable design that protects user data both from unauthorized use and from corruption or loss. A transaction is an atomic grouping of metadata management functions such that either all of them take place together or none of them takes place. Journaling makes it possible to back out any partially complete transactions if a failure occurs. Transaction technology is common in relational data management systems but not in storage systems. It is the key to maintaining reliability and security while scaling upward into a large, distributed storage environment.

2.2.6. Multiple Hierarchies and Classes of Services

Most other storage management systems support simple storage hierarchies consisting of one kind of disk and one kind of tape. HPSS provides multiple configurable hierarchies, which are particularly useful when inserting new storage technologies over time. As new disks or tapes are added, new classes of service can be set up. HPSS files reside in a particular class of service which users select based on parameters such as file size and performance. A class of service is implemented by a storage hierarchy which in turn consists of multiple storage classes, as shown in Figure 2. Storage classes are used to logically group storage media to provide storage for HPSS files. A hierarchy may be as simple as a single tape, or it may consist of two or more levels of disk and local tape. The user can even set up classes of service so that data from an older type of tape is subsequently migrated to a new type of tape. Such a procedure allows migration to new media over time without having to copy all the old media at once.

2.2.7. Storage Subsystems

Storage subsystems (or simply, "subsystems") may be used to increase the scalability of HPSS in handling concurrent requests or to meet local political needs. Each subsystem contains a single Core Server. If migration and purge are needed for the subsystem, then it will also contain a Migration / Purge Server. In addition, if HPSS is to be used as a backing store for a 'linked' filesystem such as XFS, a DMAP Gateway will be required. All other servers are subsystem-independent.

Data stored within HPSS is assigned to different subsystems based on pathname resolution. A pathname consisting of '/' resolves to the root Core Server which is specified in the global configuration file. However, if the pathname contains junction components, it may resolve to a Core Server in a different subsystem. For example, the pathname '/JunctionToSubsys2/mydir' could lead to a fileset managed by the Core Server in subsystem 2. Sites which do not wish to partition their HPSS through the use of subsystems will run HPSS with a single subsystem.

2.3. HPSS Components

The components of HPSS include files, filesets, junctions, virtual volumes, physical volumes, storage segments, metadata, servers, infrastructure, user interfaces, a management interface, and policies. Media and file metadata are represented by data structures that describe the attributes and characteristics of storage system components such as files, filesets, junctions, storage segments, and volumes. Servers are the processes that control the logic of the system and control movement of the data. The HPSS infrastructure provides the services that are used by all the servers for operations such as sending messages and providing reliable transaction management. User interfaces provide several different views of HPSS to applications with different needs. The management interface provides a way to administer and control the storage system and implement site policy.

These HPSS components are discussed below in Sections 2.3.1 through 2.3.7.

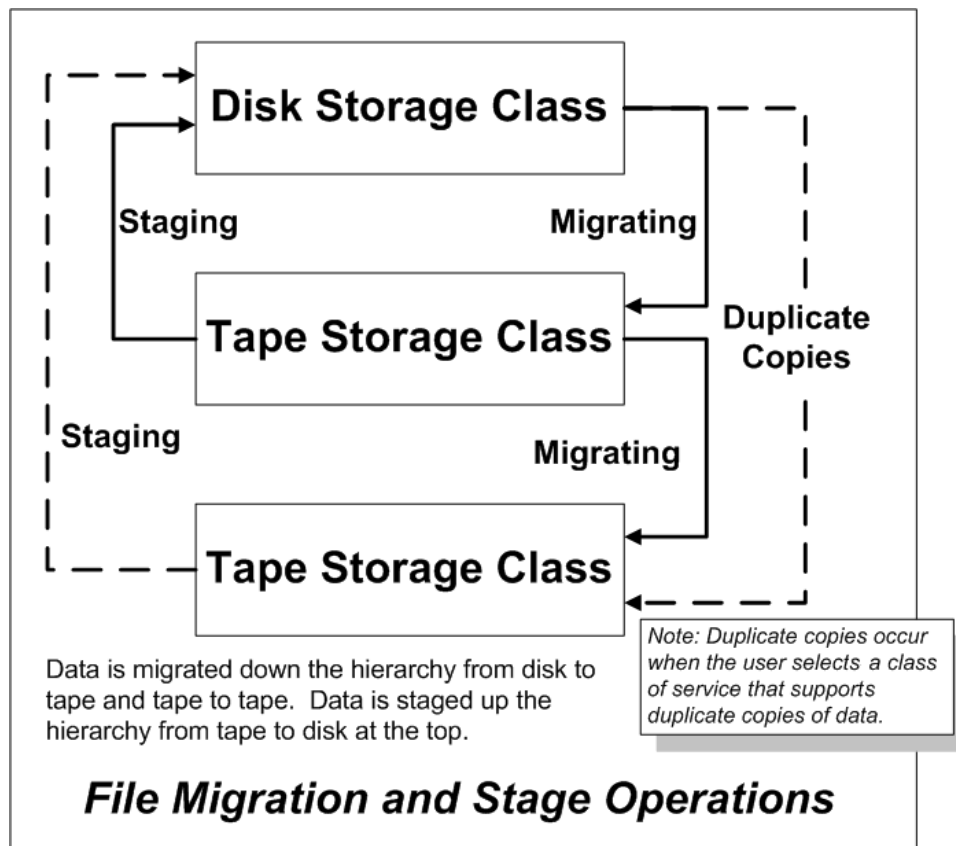


Figure 1. File Migration and Stage Operations

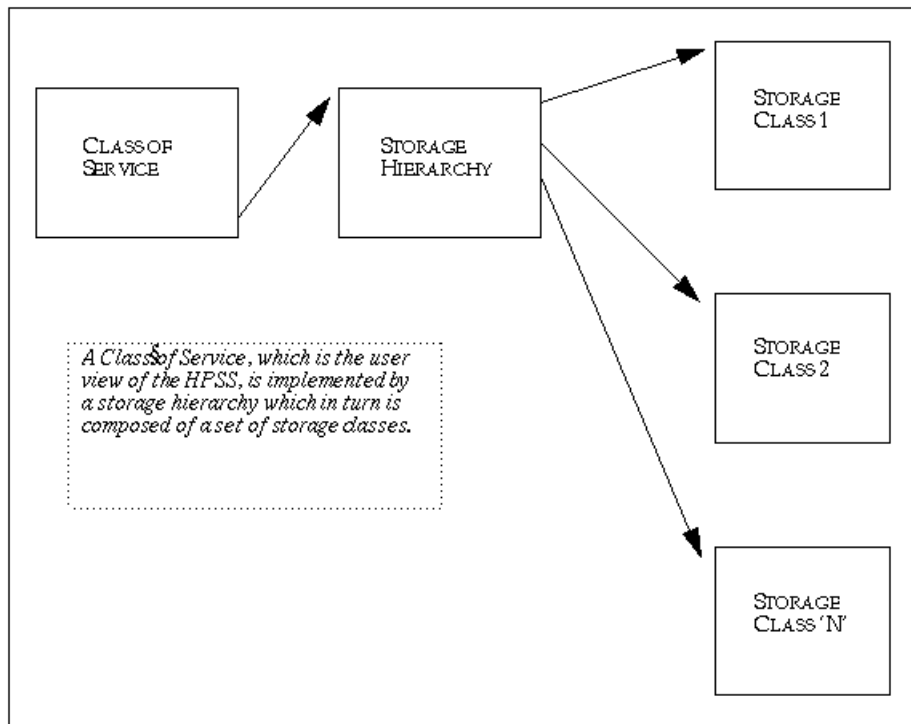


Figure 2. Class of Service / Hierarchy / Storage Class

2.3.1. HPSS Files, Filesets, Volumes, Storage Segments and Related Metadata

The various metadata constructs used to describe the HPSS namespace and HPSS storage are described below:

- **Files (Bitfiles).** Files in HPSS, called bitfiles in deference to IEEE Mass Storage Reference Model terminology, are logical strings of bytes, even though a particular bitfile may have a structure imposed by its owner. This unstructured view decouples HPSS from any particular file management system that host clients of HPSS might use. HPSS bitfile size is limited to $2^{64} - 1$ bytes.

Each bitfile is identified by a machine-generated name called a bitfile ID. It may also have a human readable name. It is the job of the HPSS Core Server (discussed in Section 2.3.2) to map a human readable name to a bitfile's ID.

- **Filesets.** A fileset is a logical collection of files that can be managed as a single administrative unit, or more simply, a disjoint directory tree. A fileset has two identifiers: a human readable name and a numeric fileset ID. Both identifiers are unique to a given realm.
- **Junctions.** A junction is a Core Server object, much like a symbolic link to a directory, that is used to point to a fileset. This fileset may belong to the same Core Server or to a different Core Server. When pointing to a different Core Server, junctions allow HPSS users to traverse to different subsystems.
- **File Families.** HPSS files can be grouped into families. All files in a given family are recorded on a set of tapes assigned to the family. Only files from the given family are

recorded on these tapes. HPSS supports grouping files on tape volumes only. Families can only be specified by associating the family with a fileset. All files created in the fileset belong to the family. When one of these files is migrated from disk to tape, it is recorded on a tape with other files in the same family. If no tape virtual volume is associated with the family, a blank tape is reassigned from the default family. The family affiliation is preserved when tapes are repacked.

- **Physical Volumes.** A physical volume is a unit of storage media on which HPSS stores data. The media can be removable (e.g., cartridge tape, optical disk) or non-removable (magnetic disk). Physical volumes may also be composite media, such as RAID disks, but must be represented by the host OS as a single device.

Physical volumes are not visible to the end user. The end user stores bitfiles into a logically unlimited storage space. HPSS, however, must implement this storage on a variety of types and quantities of physical volumes.

For a list of the tape physical volume types supported by HPSS, see Table 2 in Chapter 3.

- **Virtual Volumes.** A virtual volume is used by the Core Server to provide a logical abstraction or mapping of physical volumes. A virtual volume may include one or more physical volumes. Striping of storage media is accomplished by the Core Servers by collecting more than one physical volume into a single virtual volume. A virtual volume is primarily used inside of HPSS, thus hidden from the user, but its existence benefits the user by making the user's data independent of device characteristics. Virtual volumes are organized as strings of bytes up to $2^{64}-1$ bytes in length that can be addressed by an offset into the virtual volume.
- **Storage Segments.** A storage segment is an abstract storage object which is mapped onto a virtual volume. Each storage segment is associated with a storage class (defined below) and has a certain measure of location transparency. The Core Server (discussed in Section 2.3.2) uses both disk and tape storage segments as its primary method of obtaining and accessing HPSS storage resources. Mappings of storage segments onto virtual volumes are maintained by the HPSS Core Servers.
- **Storage Maps.** A storage map is a data structure used by the Core Server to manage the allocation of storage space on virtual volumes.
- **Storage Classes.** A storage class defines a set of characteristics and usage parameters to be associated with a particular grouping of HPSS virtual volumes. Each virtual volume and its associated physical volumes belong to a single storage class in HPSS. Storage classes in turn are grouped to form storage hierarchies (see below). An HPSS storage class is used to logically group storage media to provide storage for HPSS files with specific intended usage, similar size and usage characteristics.
- **Storage Hierarchies.** An HPSS storage hierarchy defines the storage classes on which files in that hierarchy are to be stored. A hierarchy consists of multiple levels of storage, with each level representing a different storage class. Files are moved up and down the hierarchy via migrate and stage operations based on usage patterns, storage availability, and site policies. For example, a storage hierarchy might consist of a fast disk, followed by a fast data transfer and medium storage capacity robot tape system, which in turn is followed by a large data storage capacity but relatively slow data transfer tape robot system. Files are placed on a particular level in the hierarchy depending upon the migration levels that are associated with each level in the hierarchy. Multiple copies are controlled by this mechanism. Also data can be placed at higher levels in the hierarchy by staging operations. The staging and migrating of data is shown in Figure 1: *File Migration and Stage Operations*.

- **Class of Service (COS).** Each bitfile has an attribute called Class Of Service. The COS defines a set of parameters associated with operational and performance characteristics of a bitfile. The COS results in the bitfile being stored in a storage hierarchy suitable for its anticipated and actual size and usage characteristics. Figure 2: *Class of Service/Hierarchy/Storage Class* shows the relationship between COS, storage hierarchies, and storage classes.

2.3.2. HPSS Servers

HPSS servers include the Core Server, Migration/Purge Server, Gatekeeper, Location Server, Log Client, Log Daemon, Physical Volume Library, Physical Volume Repository, Mover, Storage System Manager, and, possibly, a Data Migration Gateway. Figure 3: *HPSS Components* provides a simplified view of the HPSS system. Each major server component is shown, along with the basic control communications paths (thin arrowed lines). Infrastructure items (those components that “glue together” the distributed servers) are shown at the top of the cube. These infrastructure items are discussed in Section 2.3.4: *HPSS Infrastructure* on page 44. HPSS user interfaces (the clients listed in the figure) are also discussed in Section 2.3.5: *HPSS User Interfaces* on page 47.

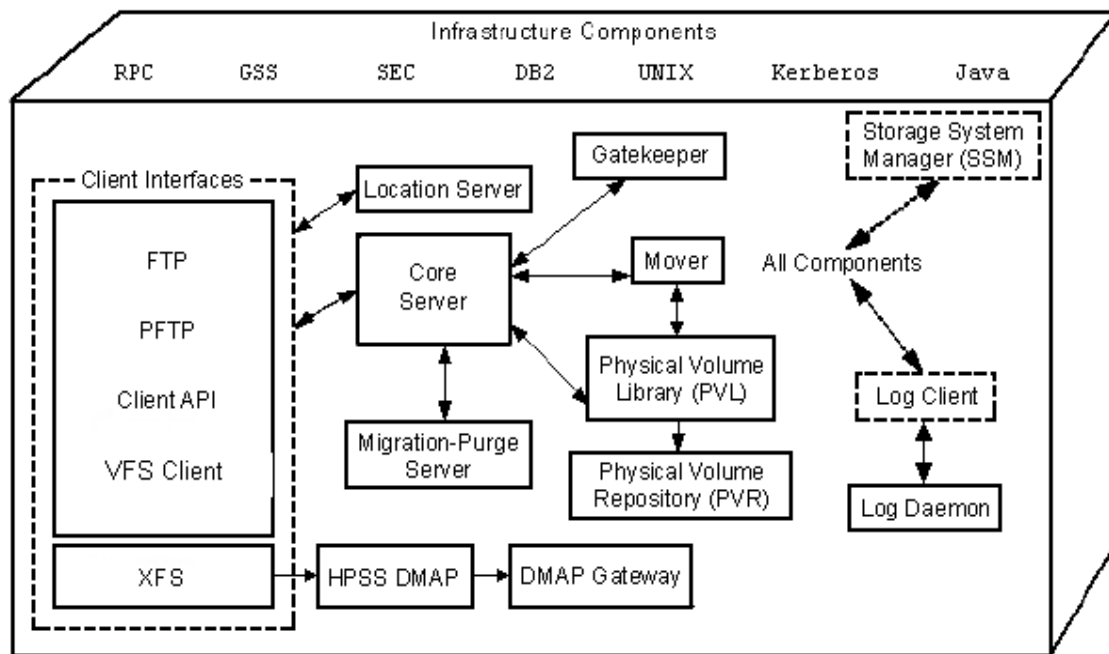


Figure 3. *HPSS Components*

- **Core Server.** The Core Server provides several key sets of functionality. First, the Core Server provides translation between human-oriented names and HPSS object identifiers. Name space objects managed by the Core Server are filesets, junctions, directories, files, hard links, and symbolic links. The Core Server provides access verification to objects and mechanisms for manipulating access to these objects via a Portable Operating System Interface (POSIX) view of the name space. This name space is a hierarchical structure

consisting of directories, files, and links. These name space objects may exist within filesets that are connected via junctions.

Second, the Core Server provides the abstraction of logical bitfiles to its clients. A bitfile is identified by a Core Server generated name called a bitfile ID. Clients may reference portions of a bitfile by specifying the bitfile ID and a starting address and length. The Core Server supports random access to files and sparsely written files. It supports parallel reading and writing of data to bitfiles and performs the mapping of logical portions of bitfiles onto physical storage devices. The Core Server supports the migration, purging, and staging of data in a storage hierarchy (though the migration/purge policies are implemented through the Migration/Purge Server, a client to the Core Server).

Third, the Core Server provides a hierarchy of storage objects: storage segments, virtual volumes, and physical volumes. The Core Server translates storage segment references into virtual volume references and then into physical volume references, handles the mapping of physical resources into striped virtual volumes to allow parallel I/O to that set of resources, and schedules the mounting and dismounting of removable media through the Physical Volume Library (see below).

- **Migration/Purge Server (MPS).** The MPS allows a site to implement its storage management policies by managing the placement of data on HPSS storage media using site-defined migration and purge policies. By making appropriate calls to its Core Server, an MPS copies data to lower levels in the hierarchy (migration), removes data from the current level once copies have been made (purge), or moves data between volumes at the same level (lateral move). Based on the hierarchy configuration, MPS can be directed to create duplicate copies of data when it is being migrated from disk or tape. This is done by copying the data to multiple lower levels in the storage hierarchy.

There are three types of migration: disk migration, tape file migration, and tape volume migration. The designation disk or tape refers to the type of storage class that migration is running against. See Section 3.7.2: *Migration/Purge Server* on page 83 for a more complete discussion of the different types of migration.

MPS runs migration on each storage class periodically using the time interval specified in the migration policy for that class. See Section 2.3.7: *HPSS Policy Modules* on page 47 for details on migration and purge policies. Migration runs can be started automatically when the warning or critical space thresholds for the storage class are exceeded. In addition, migration runs can be started manually by an administrator.

Purge runs are started automatically on each storage class when the free space in that class falls below the percentage specified in the purge policy. Purge runs may also be started manually.

Disk Migration/Purge:

The purpose of disk migration is to make one or more copies of disk files to lower levels in the hierarchy. The number of copies depends on the configuration of the hierarchy. For disk, migration and purge are separate operations. It is common for disk storage class which have been configured for migration to also be configured for purge as well. Once a file has been migrated (copied) downwards in the hierarchy, it becomes eligible for purge, which subsequently removes the file from the higher level and allows the disk space to be reused.

Tape File Migration:

The purpose of tape file migration is to make an additional copy (or multiple additional copies) of a file, in a tape storage class, to a lower level in the hierarchy. It is also possible to move files downwards instead of copying them. In this case there is no duplicate copy maintained. There is no separate purge component to tape file migration. Empty volumes must be reclaimed using the reclaim utility.

Tape Volume Migration:

The purpose of tape volume migration is to free tape volumes for reuse. Tape volumes are selected based on being in the EOM map state and containing the most unused space (caused by users overwriting or deleting files). The remaining segments on these volumes are either migrated downwards to the next level in the hierarchy, or are moved laterally to another tape volume at the same level. This results in empty tape volumes which may then be reclaimed. Note that there is no purge component to tape volume migration. All of the operations use a move instead of a copy semantic.

- **Gatekeeper (GK).** The Gatekeeper provides two main services:
 - It provides sites with the ability to schedule the use of HPSS resources using the Gatekeeping Service.
 - It provides sites with the ability to validate user accounts using the Account Validation Service.

Both of these services allow sites to implement their own policy.

The default Gatekeeping Service policy is to not do any gatekeeping. Sites may choose to implement a policy for monitoring authorized callers, creates, opens, and stages. The Core Server will call the appropriate GK API depending on the requests that the site-implemented policy is monitoring.

The Account Validation Service performs authorizations of user storage charges. A site may perform no authorization, default authorization, or site-customized authorization depending on how the Accounting Policy is set up and whether or not a site has written site-specific account validation code. Clients call this service when creating files, changing file ownership, or changing accounting information. If Account Validation is enabled, the Account Validation Service determines if the user is allowed to use a specific account or gives the user an account to use, if needed. The Core Server also calls this service to perform an authorization check just before account-sensitive operations take place.

- **Location Server (LS).** The Location Server acts as an information clearinghouse to its clients through the HPSS Client API to enable them to locate servers and gather information from both local and remote HPSS systems. Its primary function is to allow a client to determine a server's location and, by knowing other information about the server such as its object UUID, determine its server type or its subsystem id. This allows a client to contact the appropriate server. Usually the Location Server is only used by the Core Server or the Gatekeeper.
- **Physical Volume Library (PVL).** The PVL manages all HPSS physical volumes. It is in charge of mounting and dismounting sets of physical volumes, allocating drive and cartridge resources to satisfy mount and dismount requests, providing a mapping of physical volume to cartridge and of cartridge to Physical Volume Repository (PVR), and issuing commands to PVRs to perform physical mount and dismount actions. A primary function of the PVL is the support for atomic mounts of sets of cartridges for parallel access to data. Atomic mounts are implemented by the PVL, which waits until all necessary cartridge resources for a request are available before issuing mount commands to the PVRs.

- **Physical Volume Repository (PVR).** PVRs manage HPSS cartridges. Though an HPSS system may contain multiple PVRs, each cartridge is managed by only one. PVRs provide APIs for clients to request cartridge mounts and dismounts and query the status of cartridges. For convenience, PVRs are often configured in one-to-one correspondence to tape libraries.

For information on the types of tape libraries supported by HPSS PVRs, see Section 3.4.2: *Robotically Mounted Tape* on page 62.

An Operator PVR is provided for cartridges not under control of a robotic library. These cartridges are mounted on a set of drives by operators.

- **Mover (MVR).** The purpose of the Mover is to transfer data from a source device to a sink device. A device can be a standard I/O device with geometry (e.g., tape, disk) or a device without geometry (e.g., network, memory). The MVR's client (typically the Core Server) describes the data to be moved and where the data is to be sent. It is the MVR's responsibility to actually transfer the data, retrying failed requests and attempting to optimize transfers. The MVR supports transfers for disk devices, tape devices and a mover protocol that can be used as a lightweight coordination and flow control mechanism for large transfers.
- **Storage System Manager (SSM).** SSM is the tool used by the system administrator to manage HPSS. It enables the administrator to configure, monitor and control the resources (servers, devices, tape libraries, and media) of HPSS in ways that conform to the management policies of a given customer site.

Monitoring capabilities include the ability to query the values of important management attributes of storage system resources and the ability to receive notifications of alarms and other significant system events. Controlling capabilities include the ability to start up and shut down servers and the ability to set the values of management attributes of storage system resources and storage system policy parameters. Additionally, SSM can request that specific operations be performed on resources within the storage system, such as adding and deleting logical or physical resources. Operations performed by SSM are usually accomplished through standard HPSS Application Program Interfaces (APIs).

SSM has three components, one of which is an HPSS server and two of which are user interface client programs. The server is:

- **SSM System Manager.** Communicates with all other HPSS components requiring monitoring or control.

The user interface clients are:

- **SSM GUI (hpssgui)** - Provides the HPSS administrator or operator the ability to configure or monitor the HPSS System through a graphical user interface.
- **SSM Command Line Interface (hpssadm)** - Provides the HPSS administrator or operator the ability to configure or monitor a subset of the HPSS system through a set of interactive or batch commands.
- **Data Migration Gateway (DMG).** The DMG is only used if a connection to a XDMS fileset is desired. If a site wishes to have HPSS act as a backing-store for a file system such as XFS, that site will need the services of a DMG. The DMG helps to coordinate the movement of data between these so-called linked filesets and HPSS.

2.3.3. HPSS Storage Subsystems

The goal of storage subsystems (or just “subsystems”) is to increase the scalability of HPSS by allowing multiple Core Servers to be used within a single HPSS system. Every HPSS system is partitioned into one or more subsystems. Each subsystem contains a single Core Server. If migration and purge are needed, then the subsystem should contain a single Migration/Purge Server. Each Core Server and each Migration/Purge Server must exist within a storage subsystem. Each subsystem may optionally be serviced by a Gatekeeper which performs site-specific user-level scheduling of HPSS storage requests or account validation. Each Gatekeeper may service multiple subsystems. If a subsystem wishes to utilize XDSM filesets, a DMAP Gateway must also be configured. Only one DMAP Gateway is needed for multiple subsystems. All other servers exist independently of storage subsystems. Sites which do not need multiple Core Servers use a single storage subsystem.

The computer that runs the Core Server for subsystem X is referred to as the “Subsystem X node”, while the computer running the Root Core Server is known as the “Root Subsystem Node”.

Each HPSS system consists of two types of DB2 databases. The global database contains subsystem-independent data, and a subsystem database contains subsystem-dependent data. An HPSS system has exactly one global database and one or more subsystem databases.

The definitions of classes of service, hierarchies, and storage classes apply to the entire HPSS system (they are subsystem-independent). All classes of service, hierarchies, and storage classes are known to all storage subsystems within HPSS. The level of resources dedicated to these entities by each storage subsystem may differ. It is possible to disable selected classes of service within given storage subsystems. Although the class of service definitions are global, if a class of service is disabled within a storage subsystem then the Core Server in that subsystem never selects that class of service.

Since the Migration/Purge Server (MPS) is contained within the storage subsystem, migration and purge operate independently in each subsystem. Each Migration/Purge Server is responsible for migration and purge for those storage class resources contained within its particular storage subsystem. Migration and purge runs are independent and are not synchronized. Migration and purge for a storage class may be configured differently for each storage subsystem. It is possible to set up a single migration or purge policy which applies to a storage class across all storage subsystems (to make configuration easier), but it is also possible to control migration and purge differently in each storage subsystem.

Similarly, storage class thresholds may be configured differently for each storage subsystem. It is possible to set up a single set of thresholds which apply to a storage class across all storage subsystems, but it is also possible to control the thresholds differently for each storage subsystem.

2.3.4. HPSS Infrastructure

The HPSS infrastructure items (see Figure 3) are those components and services used by the various HPSS servers. The HPSS infrastructure components common among servers are discussed below.

- **Remote Procedure Calls (RPC).** Most HPSS servers, with the exception of the MVR, PFTPD, and logging services (see below), communicate requests and status (control information) via RPCs. HPSS does not use RPCs to move user data. RPCs provide a communication interface resembling simple, local procedure calls.
- **Thread Services.** HPSS uses a threads package for multitasking. The threads package is vital for HPSS to serve large numbers of concurrent users and to enable multiprocessing of its servers.
- **Transaction Management.** Requests to perform actions, such as creating bitfiles or accessing file data, result in client-server interactions between software components. The

HPSS Core Server performs most of the HPSS metadata changes using the transaction management tools provided by DB2. For the most part, these metadata transactions are managed entirely within the Core Server. Other servers such as MPS and PVL modify their metadata transactionally, and those transactions are entirely contained within those servers. A very small number of rarely performed operations require distributed transaction management, and these are handled by DB2 as well.

Transactional integrity to guarantee consistency of server state and metadata is required in HPSS in case a particular component fails. HPSS metadata updates utilize the transactional capability of DB2. The selection of DB2 was based on functionality and vendor platform support. It provides HPSS with an environment in which a job or action completes successfully or is aborted completely.

DB2 provides a full suite of recovery options for metadata transactions. Recovery of the database to a consistent state after a failure of HPSS or DB2 is automatic. A full suite of database backup and maintenance tools is provided as well.

- **Security.** HPSS security software provides mechanisms that allow HPSS components to communicate in an authenticated manner, to authorize access to HPSS objects, to enforce access control on HPSS objects, and to issue log records for security-related events. The security components of HPSS provide authentication, authorization, enforcement, and audit capabilities for the HPSS components. Customer sites may use the default security policy delivered with HPSS or define their own security policy by implementing their own version of the security policy module.
 - *Authentication* — is responsible for guaranteeing that a principal (a customer identity) is the entity that is claimed, and that information received from an entity is from that entity.
 - *Authorization* — is responsible for enabling an authenticated entity access to an allowed set of resources and objects. Authorization enables end user access to HPSS directories and bitfiles.
 - *Enforcement* — is responsible for guaranteeing that operations are restricted to the authorized set of operations.
 - *Audit* — is responsible for generating a log of security-relevant activity. HPSS audit capabilities allow sites to monitor HPSS authentication, authorization, and file security events. File security events include file creation, deletion, opening for I/O, and attribute modification operations.

HPSS components that communicate with each other maintain a joint security context. The security context for both sides of the communication contains identity and authorization information for the peer principals as well as an optional encryption key.

Access to HPSS server interfaces is controlled through an Access Control List (ACL) mechanism. Membership on this ACL is controlled by the HPSS administrator.

- **Logging.** A logging infrastructure component in HPSS provides an audit trail of server events. Logged data includes alarms, events, requests, security audit records, status records, and trace information. The Log Client, which may keep a temporary local copy of logged information, communicates log messages to a central Log Daemon, which in turn maintains a central log. Depending on the type of log message, the Log Daemon may send the message to the SSM for display purposes. When the central HPSS log fills, messages are sent to a secondary log file. A configuration option allows the filled log to be automatically archived to

HPSS. A delog function is provided to extract and format log records from a central or archived log file. Delog options support filtering by time interval, record type, server, and user.

- **Accounting.** The HPSS accounting system provides the means to collect usage information in order to allow a particular site to charge its users for the use of HPSS resources. It is the responsibility of the individual site to sort and use this information for subsequent billing based on site-specific charging policies. For more information on the HPSS accounting policy, refer to Section 2.3.7: *HPSS Policy Modules* on page 47.

2.3.5. HPSS User Interfaces

As indicated in Figure 3, HPSS provides the user with a number of transfer interfaces as discussed below.

- **File Transfer Protocol (FTP).** HPSS provides an industry-standard FTP user interface. Because standard FTP is a serial interface, data sent to a user is received serially. This does not mean that the data within HPSS is not stored and retrieved in parallel; it means that the Parallel FTP Daemon within HPSS must consolidate its internal parallel transfers into a serial data transfer to the user. HPSS FTP performance in many cases will be limited not by the speed of a single storage device but by the speed of the data path between the HPSS Parallel FTP Daemon and the user's FTP client.
- **Parallel FTP (PFTP).** The PFTP supports standard FTP commands plus extensions and is built to optimize performance for storing and retrieving files from HPSS by allowing data to be transferred in parallel across the network media. The parallel client interfaces have a syntax similar to FTP but with numerous extensions to allow the user to transfer data to and from HPSS across parallel communication interfaces established between the PFTP client and the HPSS Movers. This provides the potential for using multiple client nodes as well as multiple server nodes. PFTP supports transfers via TCP/IP. The PFTP client establishes a **control** connection with the HPSS Parallel FTP Daemon and subsequently establishes TCP/IP **data** connections directly with HPSS Movers to transfer data at rates limited only by the underlying media, communications hardware, and software.
- **Client Application Program Interface (Client API).** The Client API is an HPSS-specific programming interface that mirrors the POSIX.1 specification where possible to provide ease of use to POSIX application programmers. Additional APIs are also provided to allow the programmer to take advantage of the specific features provided by HPSS (e.g., storage/access hints passed on file creation and parallel data transfers). The Client API is a programming level interface. It supports file open/create and close operations; file data and attribute access operations; file name operations; directory creation, deletion, and access operations; and working directory operations. HPSS users interested in taking advantage of parallel I/O capabilities in HPSS can add Client API calls to their applications to utilize parallel I/O. For the specific details of this interface see the *HPSS Programmer's Reference Guide*, Volume 1.
- **HPSS VFS Interface.** The HPSS VFS Interface presents a standard POSIX I/O interface to a user application. This obviates the need for a user application to be rewritten against the HPSS Client API and hence can be used "out of the box" as long as the user application is POSIX compliant. A portion of an HPSS directory tree can be mounted on a client machine as if it were a local POSIX-compliant filesystem.

2.3.6. HPSS Management Interfaces

HPSS provides a graphical user interface, the SSM hpssgui, for HPSS administration and operations

GUI. The hpssgui simplifies the management of HPSS by organizing a broad range of technical data into a series of easy-to-read graphic displays. The hpssgui allows monitoring and control of virtually all HPSS processes and resources from windows that can easily be added, deleted, moved, or overlapped as desired.

HPSS also provides a command line SSM interface, hpssadm. This tool does not provide all the functionality of the hpssgui, but does implement a subset of its frequently used features, such as some monitoring and some control of servers, devices, storage classes, volumes, and alarms. It is useful for performing HPSS administration from remote locations where network traffic is slow or difficult. Additionally, hpssadm provides some rudimentary mass configuration support by means of the ability to issue configuration commands from a batch script.

In addition to SSM, HPSS provides a number of command line utilities for specialized management purposes, such as listing the volumes managed by a particular PVR or core server. See the *HPSS Management Guide Chapter 14: Management Tools* for more information. See also the HPSS man pages for descriptions of these utilities.

2.3.7. HPSS Policy Modules

There are a number of aspects of storage management that probably will differ at each HPSS site. For instance, sites typically have their own guidelines or policies covering the implementation of accounting, security, and other storage management operations. In order to accommodate site-specific policies, HPSS has implemented flexible interfaces to its servers to allow local sites the freedom to tailor management operations to meet their particular needs.

HPSS policies are implemented using two different approaches. Under the first approach, used for migration, purge, and logging policies, sites are provided with a large number of parameters that may be used to implement local policy. Under the second approach, HPSS communicates information through a well-defined interface to a policy software module that can be completely replaced by a site. Under both approaches, HPSS provides a default policy set for users.

- **Migration Policy.** The migration policy defines the conditions under which data is copied from one level in a storage hierarchy to one or more lower levels. Each storage class that is to have data copied from that storage class to a lower level in the hierarchy has a migration policy associated with it. The MPS uses this policy to control when files are copied and how much data is copied from the storage class in a given migration run. Migration runs are started automatically by the MPS based upon parameters in the migration policy.

Note that the number of copies which migration makes and the location of these copies is determined by the definition of the storage hierarchy and not by the migration policy.

- **Purge Policy.** The purge policy defines the conditions under which data that has already been migrated from a disk storage class can be deleted. Purge applies only to disk storage classes. It is common, but not necessary, for disk storage classes which have a migration policy to also have a purge policy. Purge runs are started automatically by the MPS based upon parameters in the purge policy.
- **Logging Policy.** The logging policy controls the types of messages to log. On a per server basis, the message types to write to the HPSS log may be defined. In addition, for each server, options to send Alarm, Event, or Status messages to SSM may be defined.
- **Security Policy.** Security policy defines the authorization and access controls to be used for client access to HPSS. HPSS security policies are provided to control access (authentication) from FTP and/or Parallel FTP using **Username/Password**, **Ident**, or **Kerberos** credentials.

HPSS provides facilities for recording information about authentication and object (file/directory) creation, deletion, access, and authorization events. The security audit policy for each server determines the records that each individual server will generate. All servers can generate authentication records.

- **Accounting Policy.** The accounting policy provides runtime information to the accounting report utility and to the Account Validation service of the Gatekeeper. It helps determine what style of accounting should be used and what level of validation should be enforced.

The two types of accounting are site-style and UNIX-style. The site-style approach is the traditional type of accounting in use by most mass storage systems. Each site will have a site-specific table (Account Map) that correlates the HPSS account index number with their local account charge codes. The UNIX-style approach allows a site to use the user identifier (UID) for the account index. The UID is passed along in UNIX-style accounting just as the account index number is passed along in site-style accounting.

Account Validation allows a site to perform usage authorization of an account for a user. It is turned on by enabling the Account Validation field of the Accounting Policy configuration screen. If Account Validation is enabled, the accounting style in use at the site is determined by the Accounting Style field. A site policy module may be implemented by the local site to perform customized account validation operations. The default Account Validation behavior is performed for any Account Validation operation that is not overridden by the site policy module.

- **Location Policy.** The location policy defines how Location Servers at a given site will perform, especially in regards to how often server location information is updated. All local, replicated Location Servers update information according to the same policy.
- **Gatekeeping Policy.** The Gatekeeper provides a Gatekeeping Service along with an Account Validation Service. These services provide the mechanism for HPSS to communicate information through a well-defined interface to a policy software module that can be written by a site. The site policy code is placed in well-defined shared libraries for the gatekeeping policy and the accounting policy (/opt/hpss/lib/libgksite.[a|so] and /opt/hpss/lib/libacctsite.[a|so] respectively) which are linked to the Gatekeeper. The Gatekeeping policy shared library contains a default policy which does NO gatekeeping. Sites will need to enhance this library to implement local policy rules if they wish to monitor and load-balance requests.

2.4. HPSS Hardware Platforms

2.4.1. Server Platforms

HPSS requires at least one AIX or Linux node for the core server components. A server node must have sufficient processing power and memory to handle the work load.

2.4.2. Client Platforms

The full-function Client API can be ported to any platform that supports UNIX.

The PFTP client code and Client API source code for platforms other than AIX and Linux are not on the HPSS distribution image. Maintenance of the PFTP and Client API software on platforms other than AIX and Linux is the responsibility of the customer, unless a support agreement is negotiated with IBM. Contact your HPSS Support Representative for information on how to obtain the needed software.

The following matrix illustrates which platforms support HPSS interfaces.

Table 1. HPSS Client Interface and Mover Platforms

Platform	PFTP Client	Client API	HPSS Mover	HPSS VFS Client	FTP Clients
IBM AIX	X	X	X		Any platform running standard FTP clients. GUI-based Clients may not function correctly for some commands.
Sun Solaris (Big Endian ONLY)	X	X	X		
Digital UNIX	X Not Tested				
Hewlett-Packard HPUX	X Not Tested				
Silicon Graphics IRIX (32-bit)	X		X		
Compaq Tru64	X Not Tested				
Linux (Intel)	X	X	X	X	
Linux (Power PC)		X		X	

2.4.3. Mover Platforms

Movers are used to control the logical network attachment of storage devices and are configured to run on one or more nodes. A Mover consists of two parts: The Mover administrative process that runs on the server node, and the remote Mover process that handles the HPSS devices and data transfers. Movers can run on AIX, Linux, IRIX, and Solaris systems. See Table 1 above for a detailed list of supported platforms.

Chapter 3. HPSS Planning

3.1. Overview

This chapter provides HPSS planning guidelines and considerations to help the administrator effectively plan, and make key decisions about, an HPSS system.



The planning process for HPSS must be done carefully to ensure that the resulting system satisfies the site's requirements and operates in an efficient manner. We recommend that the administrator read this entire chapter before planning the system.

The following paragraphs describe the recommended planning steps for the HPSS installation, configuration, and operational phases.

3.1.1. HPSS System Architecture

Before getting into the details of storage sizing, it is best to understand the overall HPSS system and how the components are arranged to build the HSM. Figure 4: *HPSS Generic Configuration* on page 52 shows the basic architecture of an HPSS system including disk and tape resources and their relationship to HPSS server nodes, Mover nodes, internal and external networks, and SAN interconnections. Specifics of this architecture for a given site are developed during the proposal and initial project planning stages of a deployment. Ideally the required space is derived from requirements gathered from the HPSS Questionnaire document, known operational constraints, transfer and transaction rates, and anticipated storage capacity needs. Often the disk and tape resources are dictated by current equipment already available and budgetary constraints on what can be purchased. Specific quantities and sizing of these resource are beyond the scope of this planning document. These are largely defined by the above inputs and negotiations during the initial planning meeting in which the systems engineering team draws from experience and similar deployments to design a working architecture that balances the end-user requirements with the potential, or actual, resources available.

HPSS Generic Configuration

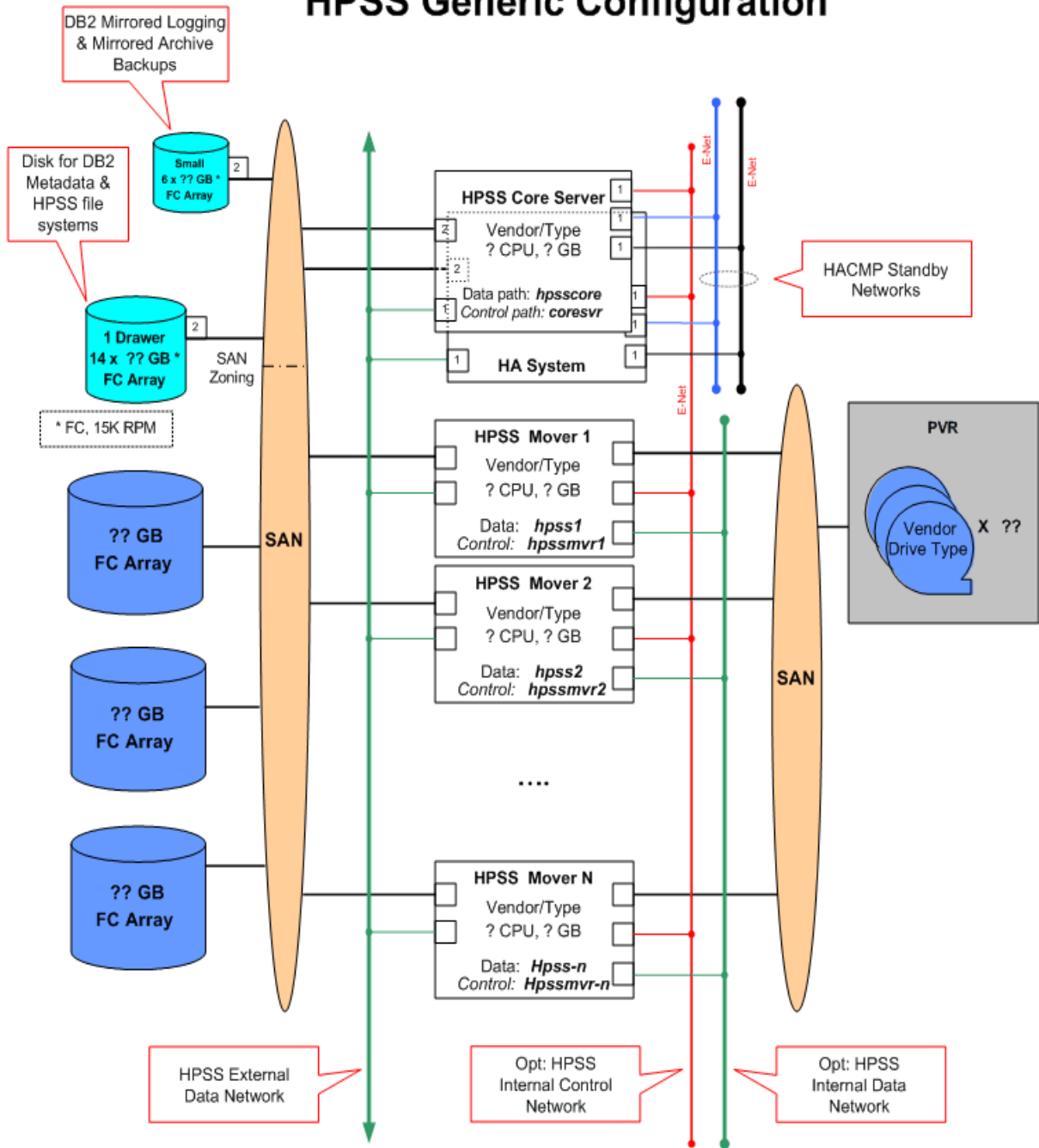


Figure 4. HPSS Generic Configuration

3.1.2. HPSS Configuration Planning

Before beginning the planning process, there is an important issue to consider. HPSS was designed to optimize the transfer of large files at the expense of some small file transfer performance. If at all possible, try to reduce the number of small files that are introduced into your HPSS system. For example, if you plan to use HPSS to backup all of the PCs in your organization, it would be best to

aggregate the individual files into large individual files before moving them into the HPSS name space.

The following planning steps must be carefully considered for the HPSS infrastructure configuration and the HPSS configuration phases:

1. Identify the site's storage requirements and policies, such as the initial storage system size, anticipated growth, usage trends, average file size, expected throughput, backup policy, and availability. For more information, see Section 3.2: *Requirements and Intended Uses for HPSS* on page 56.
2. Define the architecture of the entire HPSS system to satisfy the above requirements. The planning should:
 - Identify the nodes to be configured as part of the HPSS system.
 - Identify the disk and tape storage devices to be configured as part of the HPSS system and the nodes/networks to which each of the devices will be attached. Storage devices can be assigned to a number of nodes to allow data transfers to utilize the devices in parallel without being constrained by the resources of a single node. This capability also allows the administrator to configure the HPSS system to match the device performance with the network performance used to transfer the data between the HPSS Movers and the end users (or other HPSS Movers in the case of internal HPSS data movement for migration and staging). Refer to Section 3.4 : *Hardware Considerations* on page 62 for more discussions on the storage devices and networks supported by HPSS.
 - Identify the HPSS subsystems to be configured and how resources will be allocated among them. Refer to Section 3.8: *Storage Subsystem Considerations* on page 94 for more discussion on subsystems.
 - Identify the HPSS servers to be configured and the node where each of the servers will run. Refer to Section 3.7: *HPSS Server Considerations* on page 80 for more discussions on the HPSS server configuration.
 - Identify the HPSS user interfaces (e.g. FTP, PFTP, Client API) to be configured and the nodes where the components of each user interface will run. Refer to Section 3.6: *HPSS Interface Considerations* on page 79 for more discussion on the user interfaces supported by HPSS.
3. Ensure that the prerequisite software has been obtained, installed, and configured properly in order to satisfy the target HPSS architecture. Refer to Section 3.3: *Prerequisite Software Considerations* on page 58 for more information on the HPSS prerequisite software requirements.
4. Determine the DB2 disk storage space needed to satisfy the requirements of the HPSS system, and verify there is sufficient free space in the file systems to meet those needs. Refer to Section 3.5.4: *HPSS Metadata Space* on page 74 for more discussions of metadata sizing requirements.
5. Verify that each of the identified nodes has sufficient resources to handle the work loads to be imposed on the node. Refer to Section 3.5.5: *System Memory and Disk Space* on page 78 for more discussions on the system resource requirements.
6. Plan the design of the HPSS storage characteristics and HPSS storage space to satisfy the site's requirements:
 - Plan for file families, if any. Refer to Section 3.10.4: *File Families* on page 112 for

more information about configuring families.

- Plan for filesets and junctions, if any. Refer to Chapter 10: *Filesets and Junctions* in the *HPSS Management Guide* for more information.
 - Plan for HPSS storage classes. Refer to Section 3.10.1: *Storage Class* on page 102 for more information on the storage class configuration.
 - Plan for HPSS storage hierarchies. Refer to Section 3.10.2: *Storage Hierarchy* on page 109 for more information on the storage hierarchy configuration.
 - Plan for HPSS classes of service. Refer to Section 3.10.3: *Class of Service* on page 110 for more information on the Class of Service configuration.
 - Plan the migration and purge policy for each storage class. Refer to Section 3.9.1: *Migration Policy* on page 94 and Section 3.9.2: *Purge Policy* on page 95 for more information.
 - Determine the amount of user data storage space needed for each storage class. Refer to Section 3.5.1: *HPSS User Storage Space* on page 69 for more information on the HPSS storage space considerations.
 - Identify the disk and tape media to be imported into HPSS.
7. Define the location policy to be used. Refer to Section 3.9.6: *Location Policy* on page 99 for more information.
 8. Define the accounting policy to be used. Refer to Section 3.9.3: *Accounting Policy and Validation* on page 96 for more information on the Accounting Policy configuration.
 9. Define the logging policy for the each of the HPSS servers. Refer to Section 3.9.5: *Logging Policy* on page 99 for more information on the Logging Policy configuration.
 10. Define the security policy for the HPSS system. Refer to Section 3.9.4: *Security Policy* on page 98 for more information on the Security Policy for HPSS.
 11. Determine if a Gatekeeper will be required. It is required if a site wants to do Account Validation or Gatekeeping. Refer to Section 3.9.3: *Accounting Policy and Validation* on page 96 and Section 3.9.7: *Gatekeeping* on page 99 for more information.

3.1.3. Purchasing Hardware and Software

It is recommended that hardware be purchased only after the HPSS configuration has been planned. Purchasing the hardware prior to the planning process may result in performance and utilization issues that could easily be avoided by advance planning.

If deciding to purchase Sun, SGI, or Linux servers for storage purposes, note that the operating system limitations will only allow a fixed number of raw devices to be configured per logical unit (disk drive or disk array). Sun's Solaris operating system currently allows only eight partitions per logical unit (one of which is used by the system) unless the optional volume manager is used. SGI's IRIX operating system currently allows only sixteen partitions per logical unit. Linux operating system limits SCSI disks to 15 partitions and limits IDE disks to 63 partitions unless LVM is used. These limits can potentially impact the utilization of a disk drive or disk array.

Refer to Section 3.5: *HPSS Sizing Considerations* on page 68 for more information on calculating the number and size of devices that will be needed to meet your requirements.

Refer to Section 3.3: *Prerequisite Considerations* on page 58 for more information on the required software that will be needed to run HPSS.

3.1.4. HPSS Operational Planning

The following planning steps must be carefully considered for the HPSS operational phase:

1. Define the site guidelines for the HPSS users and SSM users.
 - Each HPSS user who uses the storage services provided by HPSS should be assigned an Accounting ID and one or more appropriate Classes of Service (COS) to store files.
 - Each SSM user (administrator or operator) should be assigned an appropriate SSM security level. The SSM security level defines what functions each SSM user can perform on HPSS through SSM. Refer to Section 2.3: *SSM User Security*, Section 3.3.2: *Creating the SSM User Accounts*, and Section 12.1.1.6: *Add an SSM User ID* of the *HPSS Management Guide* for more information on setting up the security level for an SSM user.
2. Define the site procedures for repacking and reclaiming HPSS tape volumes. Define the tape consolidation and reuse goals. For instance, define a tape utilization factor and plan to repack those tapes that fall below that limit. The limits can be set on a per Storage Class basis. Also decide when, or if empty tapes will be reclaimed. Refer to Section 8.1.4: *Repacking Tape Virtual Volumes* and Section 8.1.5: *Reclaiming HPSS Tape Virtual Volumes* (both in the *HPSS Management Guide*) for more information.
3. Define the site policy and procedure for generating the accounting reports. Take into consideration how often an accounting report needs to be generated, how the accounting information from the report will be used to produce the desired cost accounting, and whether the accounting reports need to be archived. Refer to Section 3.9.3: *Accounting Policy and Validation* on page 96 and Section 12.2: *Accounting Policy* of the *HPSS Management Guide* for more information on defining an Accounting Policy and generating accounting reports.
4. Determine if gatekeeping (monitoring or load-balancing) will be required. If so, define and write the site policy code for gatekeeping. Refer to Section 3.9.7: *Gatekeeping* on page 99 for more information on gatekeeping, and refer to the *HPSS Programmers Reference* for guidelines on implementing the Site Interfaces for the Gatekeeping Service.

3.1.5. HPSS Deployment Planning

The successful deployment of an HPSS installation is a complicated task which requires reviewing client/system requirements, integration of numerous products and resources, proper training of users/administrators, and extensive integration testing in the client environment.

Consider first, creating a set of documents to ensure the resources and intended configuration of those resources can adequately meet the expectations required of the system. The next step in this process is to coordinate the availability and readiness of the resources before the actual installation of HPSS begins. Each one of the products/resources that HPSS uses must be installed, configured and tuned for the final system to function and perform as expected. Once installed, a series of tests should be planned and performed to verify that the system can meet the demands of the final production environment. Finally, proper training of those administrating the system, as well as those who will use it, is necessary to make a smooth transition to production usage.

To help the HPSS system administrators in all of these tasks, a set of procedures have been developed to supplement this document. The HPSS Deployment Process contains a detailed outline of what is required to bring up an HPSS system, from an initial introduction and review of the environment to production use. This document is provided to customers at the initial HPSS planning meeting. The deployment procedures include a time line plus checklist that the HPSS customer installation/system administration team should use to keep the deployment of an HPSS system on track. This is the same

guide that the HPSS support/deployment team uses to monitor and check the progress of an installation.

3.2. Requirements and Intended Uses for HPSS

This section provides some guidance for the administrator to identify the site's requirements and expectations of HPSS. Issues such as the amount of storage needed, access speed and data transfer speed, typical usage, security, expected growth, data backup, and conversion from an old system must be factored into the planning of a new HPSS system.

3.2.1. Storage System Capacity

The amount of HPSS user data storage space the administrator must plan for includes the following considerations:

- The amount of user data storage space required to support new user accounts.
- The amount of user data storage space required to support expected growth of current accounts.
- The amount of metadata storage space required to support storage management activities such as migration and repack.
- The amount of user data storage space required to support duplicate copies of user files.

Another component of storage space planning is the amount of space needed for HPSS system metadata. Refer to Section 3.5.1: *HPSS User Storage Space* on page 69 and Section 3.5.4: *HPSS Metadata Space* on page 74 for more information on determining the needed storage space and metadata space.

3.2.2. Required Throughputs

Determine the required or expected throughput for the various types of data transfers that users will perform. Some users want quick access to small amounts of data. Other users have huge amounts of data they want to transfer quickly, but are willing to wait for tape mounts, etc. In all cases, plan for peak loads that can occur during certain time periods. These findings must be used to determine the type of storage devices and network to be used with HPSS to provide the needed throughput.

3.2.3. Load Characterization

Understanding the kind of load users are putting on an existing file storage system provides input that can be used to configure and schedule the HPSS system. What is the distribution of file sizes? How many files and how much data is moved in each category? How does the load vary with time (e.g., over a day, week, month)? Are any of the data transfer paths saturated?

Having this storage system load information helps to configure HPSS so that it can meet the peak demands. Also based on this information, maintenance activities such as migration, repack, and reclaim can be scheduled during times when HPSS is less busy.

3.2.4. Usage Trends

To configure the system properly the growth rates of the various categories of storage, as well as the growth rate of the number of files accessed and data moved in the various categories must be known. Extra storage and data transfer hardware must be available if the amount of data storage and use are growing rapidly.

3.2.5. Duplicate File Policy

The policy on duplicating user data files impacts the amount of data stored and the amount of data moved. If all user files are duplicated, the system will require twice as much tape storage. If users perform their own duplication of files, the system may consume a smaller amount of storage space. Users can be given control over duplication of their files by allowing them a choice between hierarchies which provide duplication and hierarchies which do not.

3.2.6. Charging Policy

HPSS does not charge users for the use of storage system resources. Instead, it collects information that a site can use to implement a charging policy.

3.2.7. Security

Authentication and authorization between HPSS servers is done through use of either UNIX or Kerberos security tools for authentication and either UNIX or LDAP for authorization services. By default, servers are authenticated using the Kerberos authentication service, and authorization information is obtained from the UNIX authorization service. The default protection level passes authentication tokens on the first remote procedure call to a server. The authentication service, authorization service, and protection level for each server can be configured to raise or lower the security of the system. Two cautions should be noted: (1) raising the protection level to packet integrity or packet privacy will require additional processing for each RPC, and (2) lowering the authentication service to none effectively removes the HPSS authentication and authorization mechanisms. Lowering the authentication service level should only be done in a trusted environment.

Each HPSS server authorizes and enforces access to its interfaces through access control lists stored in the AUTHZACL table. To modify server state, control access is required. Generally, this is only given to the Kerberos principal associated with the HPSS system administrative component. Additional Kerberos principals can be allowed or denied access by setting permissions appropriately. See Section 2.1: *HPSS Server Security ACLs* of the *HPSS Management Guide* for more information.

Security auditing in each server may be configured to record all, none, or some security events. Some sites may choose to log every client connection; every bitfile creation, deletion, and open; and every file management operation. Other sites may choose to log only errors. See the security information fields in the general server configuration (Section 5.2: *Server Configuration* of the *HPSS Management Guide*) for more details.

User access to HPSS interfaces depends on the interface being used. Access through the native Client API uses the UNIX or Kerberos authentication services and UNIX or LDAP authorization services described above. FTP or Parallel FTP access may utilize the HPSS password file, a configurable password file, or the Kerberos credentials. Additional FTP access is available using Ident, or Kerberos GSS credentials. Refer to the FTP section of the *HPSS User's Guide* for additional details.

3.2.7.1. Cross Realm Access

Kerberos provides facilities for secure communication between multiple Kerberos Realms (domains) referred to as Trusted "Cross Realm" access. These features use the Kerberos facilities to provide a trusted environment between cooperating locations. HPSS uses the Kerberos Cross Realm features for authentication. The procedures for inter-connecting Kerberos Realms are outlined in Section 1.5.3: *Cross Realm Cookbook* of the *HPSS Management Guide*. The HPSS Parallel FTP program can utilize the Kerberos and HPSS Cross Realm access features.

The Generic Security Service (GSS) FTP, available from the Massachusetts Institute of Technology, and the HPSS Parallel FTP applications can take advantage of the Cross Realm access features for

authentication and authorization (subject to certain caveats – See FTP documentation for details). The **pftp_client** binary must be built using the distributed source code. However, it is the site's responsibility to obtain the necessary Kerberos components.

ACLs entries in the AUTHZACL table and/or ACLs on HPSS directories and files may need to be added for appropriate foreign_user and/or foreign_group entries.

3.2.8. High Availability Option

The High Availability component allows HPSS to inter-operate with IBM's HACMP for AIX software. When configured with the appropriate redundant hardware, it allows failures of individual system components (network adapters, hard disks, core server nodes, power supplies, etc.) to be overcome, allowing the system to resume servicing requests with minimal downtime.

The specifics of each HA system can vary greatly such that no single formula applies to all situations. However, additional hardware will be required for each HA system (e.g., redundant core server nodes, extra core server hard disks, RS232 cables, etc.). Also, HACMP for AIX software is required.

A highly available (HA) HPSS system requires additional systems engineering and testing. It is available as a special bid item for installing and configuring HPSS on a High Availability system.

For further information on HA HPSS, please contact your HPSS Customer Service Representative.

For more information on HACMP for AIX, please see the HACMP for AIX 5L web site at: <http://www-03.ibm.com/systems/p/software/hacmp.html>.

3.3. Prerequisite Software Considerations

This section defines the prerequisite requirements for HPSS. Some products must be obtained separately from HPSS and installed prior to the HPSS installation and configuration.

3.3.1. Prerequisite Software Overview

This section describes the prerequisite software packages required by HPSS and provides information to obtain them. Refer to Section 3.3.2: *Prerequisite Summary By HPSS Node Type* on page 59 for details on software versions.

3.3.1.1. DB2

HPSS uses the DB2 Universal Database Enterprise Server Edition by IBM Corporation to manage all HPSS metadata. DB2 software is included in the HPSS distribution. Refer to Section 5.3.1.2: *Install HPSS Documentation and DB2 Software* on page 141 for more information. The required DB2 FixPak must be downloaded and installed after the DB2 base is installed. DB2 FixPaks can be downloaded from the DB2 Universal Database for Linux, UNIX, and Windows webpage at <http://www-306.ibm.com/software/data/db2/udb/support/downloadadv8.html>.

3.3.1.2. Kerberos

HPSS uses Massachusetts Institute of Technology (MIT) Kerberos to implement Kerberos authentication. MIT Kerberos is a network authentication protocol designed to provide authentication for client/server applications by using secret-key cryptography. A free implementation of this protocol can be downloaded from the MIT's website (<http://web.mit.edu/kerberos/>). Refer to Section 5.2.2: *Install MIT Kerberos (If Using Kerberos Authentication)* on page 137 for more information.

For Linux, Kerberos is installed as part of the operating system.

If UNIX authentication will be used, this product is not required.

3.3.1.3. LDAP and IBM Kerberos

HPSS can be configured to use an LDAP directory to store its authorization information such as users' names, UIDs, GIDs, and home directories. The supported LDAP server product for this release is IBM Tivoli Directory Server. It can be downloaded from the <http://www-306.ibm.com/software/tivoli/resource-center/security/code-directory-server.jsp> webpage.

Installing the IBM Kerberos client is only required if LDAP is being used for authorization and the LDAP daemon will be used for Kerberos authentication. This option is supported only on AIX. The fileset for the IBM Kerberos client is located on the AIX Expansion Pack CD.

If UNIX authorization will be used, this product is not required.

3.3.1.4. Java

HPSS uses the Java 2 Standard Edition, version 1.4.2, to implement the SSM graphical user interface, **hpssgui**, the SSM command line interface, **hpssadm** and the **mkhpss** utility.

The Java product required for the AIX platform can be downloaded from the IBM Developer Kits for AIX, Java Technology Edition webpage,

<http://www-106.ibm.com/developerworks/java/jdk/aix.index.html>.

The Java product required for the Linux and Windows platform can be downloaded from Sun Microsystems' download webpage, <http://java.sun.com/j2se/1.4.2/download.html>.

3.3.2. Prerequisite Summary By HPSS Node Type

This section provides a summary list of prerequisite software required for HPSS. It also lists the software versions which have been verified with HPSS 6.2.

3.3.2.1. HPSS Server Nodes

This section describes the prerequisite software required for each server node.

3.3.2.1.1. AIX Requirements

Each AIX server node must have the following installed:

- IBM RS/6000 (eServer pSeries) with a minimum of 2 GB RAM
- AIX 5.2 with ML9 + APAR IY89387 or AIX 5.3 with ML5 + APAR IY89429 (32-bit or 64-bit)
- DB2 UDB V8.2 Enterprise Server Edition (ESE) for AIX with FixPak 4. Note that FixPak 12 for DB2 8.1 is the same as DB2 UDB Version 8.2 with FixPak 4.
- Java JDK 1.4.2 for AIX. Upgrade to at least IBM Java build ca142-20060421 (SR 5) which is APAR IY84053.
- MIT Kerberos 1.3.5 (if planning to use Kerberos authentication)
- IBM LDAP 5.1 (if planning to use LDAP authorization)
- IBM Kerberos 1.3 (if planning to use Kerberos authentication with LDAP)
- C compiler for AIX, version 7.0.0.8 (if planning to recompile HPSS code on this node)

- IBM ATL: atldd.driver 6.5.2.0 and Atape.driver 10.2.8.0 (if planning to control the IBM tape library and drives from this node)

3.3.2.1.2. Linux Requirements

Each Linux server node must have the following installed:

- Linux machine (eServer zSeries) with a minimum of 2 GB RAM
- Red Hat Enterprise Linux AS release 4 (Nahant Update 3, kernel 2.6.9-34.ELsmp)
- DB2 UDB V8.2 Enterprise Server Edition (ESE) for Linux (32-bit) with ixPak 4. Note that FixPak 12 for DB2 8.1 is the same as DB2 UDB Version 8.2 with FixPak 4.
- Java JDK 1.4.2_06 for Linux. Upgrade to J2SE 1.4.2_06 or later within the Java 1.4.2 set of fixes/releases.
- MIT Kerberos 1.3.4-10 (if planning to use Kerberos authentication - installed as part of the operating system)
- IBM LDAP 5.2 (if planning to use LDAP authorization)
- C compiler for Linux: gcc-3.4.5 (if planning to recompile HPSS code on this node)
- IBM ATL: ibmatl-5.3.9.0-0 (if planning to use IBM tape library and drives from this node)

3.3.2.2. HPSS Mover Nodes

A Mover consists of two processes: the mover administrative process that runs on the server node, and the remote mover process that handles the HPSS devices and data transfers. To maximize performance, the remote mover should not be placed on a node with DB2 and HPSS subsystem servers.

Since HPSS security, logging and metadata services are performed on the server node, no additional software, like DB2 or HPSS servers, need to be installed on the remote Mover node.

3.3.2.2.1. AIX Requirements

Each AIX Mover node must have the following prerequisites:

- IBM RS/6000 (eServer pSeries) with a minimum of 1 GB RAM
- AIX 5.2 with ML9 + APAR IY89387 or AIX 5.3 with ML5 + APAR IY89429 (32-bit or 64-bit)
- C compiler for AIX, version 7.0.0.8 (if planning to recompile HPSS code)
- IBM ATL: ibmatl-6.5.2.0 and Atape.driver 9.1.0.0 (if planning to use IBM tape library and drives from this node)

3.3.2.2.2. Linux Requirements

Each Linux Mover node must have the following prerequisites:

- Linux machine (eServer zSeries) with a minimum of 1 GB RAM
- Red Hat Enterprise Linux AS release 4 (Nahant Update 3, kernel 2.6.9-34.ELsmp)
- IBM ATL: ibmatl-6.5.2.0 (if planning to use IBM tape library from this node)

- 1 GB RAM

3.3.2.2.3. Solaris Requirements

Each Solaris Mover node must have the following prerequisites:

- Solaris UltraSPARC based processor
- Solaris 8+ (32-bit or 64-bit)
- C compiler: Forte Developer 7 C 5.4 2002/03/09 (if planning to recompile Mover code)

3.3.2.2.4. IRIX Requirements

Each IRIX Mover node must have the following prerequisites:

- SGI machine
- IRIX 6.5 (with latest/recommended patch set)
- C compiler (if planning to recompile Mover code)

3.3.2.3. HPSS Client Nodes

This section describes the prerequisite requirements for running HPSS clients.

3.3.2.3.1. SSM Client Requirements

The client node where the SSM **hpssgui** and **hpssadm** applications run must meet the following requirements:

- Supported platforms: AIX , Linux, Windows
- J2RE or J2SE 1.4.2 for the appropriate platforms

3.3.2.3.2. Client API Requirements

The client node where HPSS Client API applications run must meet the following requirements:

- Supported platforms. Refer to Table 1: *HPSS Client Interface and Mover Platforms* for complete list.
- C compiler (if planning to recompile client application)

3.3.2.3.3. FTP/PFTP Client Requirements

The client node where HPSS FTP and PFTP run must meet the following requirements:

- Supported platforms. Refer to Table 1: *HPSS Client Interface and Mover Platforms* for complete list.
- C and yacc compilers (if planning to recompile client code)

3.3.2.4. HPSS HDM Nodes (Linux only)

- SUSE Linux Enterprise Server (SLES) version 9 Service Pack 2 or greater
- MIT Kerberos 1.3.5
- dmapi-2.2.1-0.2 or higher

- xfsdump-2.2.25-0.2 or higher
- xfsprogs-2.6.25-0.2 or higher

3.4. Hardware Considerations

This section describes the hardware infrastructure needed to operate HPSS and includes considerations about infrastructure installation and operation that may impact HPSS.

3.4.1. Network Considerations

Because of its distributed nature and high-performance requirements, an HPSS system is highly dependent on the networks providing connectivity among the HPSS servers and clients.

For control communications (i.e., all communications except the actual transfer of data) among the HPSS clients and servers, HPSS requires TCP/IP services. Since control requests and replies are relatively small in size, a low-latency network usually is well suited to handling the control path.

The data path is logically separate from the control path and may also be physically separate, although this is not required. For the data path, HPSS supports the same TCP/IP networks as those supported for the control path. For supporting large data transfers, the latency of the network is less important than the overall data throughput.

HPSS also supports a special data path option that may indirectly affect network planning because it may off-load or shift some of the networking load. This option uses the shared memory data transfer method, which provides for intra-node transfers between either Movers or Movers and HPSS clients via a shared memory segment.

Along with shared memory, HPSS also supports a Local File Transfer data path for client transfers that involve HPSS Movers that have access to the client's file system. In this case, the HPSS Mover can be configured to transfer the data directly to or from the client's file.

3.4.2. Robotically Mounted Tape

All HPSS PVRs are capable of sharing a robot with other tape management systems but care must be taken when allocating drives among multiple robot users. If it is necessary to share a drive between HPSS and another tape management system, the drive can be configured in the HPSS PVR but left in the LOCKED state until it is needed. When needed by HPSS, the drive should be set to UNLOCKED and should not be used by any other tape management system while in this state. This is critical because HPSS periodically polls all of its unlocked drives even if they are not currently mounted or in use.

Generally, only one HPSS PVR is required per robot. However, it is possible for multiple PVRs to manage a single robot in order to provide drive and tape partitions within a robot. The drives in the robot must be partitioned among the PVRs and no drive should be configured in more than one PVR. Each tape is assigned to exactly one PVR when it is imported into the HPSS system and will only be mounted in drives managed by that PVR.

The tape libraries supported by HPSS are:

- IBM 3494
- IBM 3582, 3583, 3584 (LTO). These libraries are now being called the IBM TS3500.
- Spectralogic T120 (LTO, SAIT)
- STK L40 (LTO)

- STK SL500 and SL8500
- STK Tape Libraries that support ACSLS
- ADIC i500
- ADIC AML (supported by special bid only)

3.4.2.1. IBM 3494

The 3494 PVR supports Ethernet and RS-232 (TTY) attached robots. If appropriately configured, multiple robots can be accessible from a single node.

3.4.2.2. Drive-Controlled LTO Libraries (IBM 3582, IBM 3583, IBM 3584, Spectralogic T120)

The IBM 3582, IBM 3583, IBM 3584, and Spectralogic T120 Tape Libraries and Robots must be attached to Linux or AIX workstation through a SCSI interface. In each case, the library shares a SCSI channel with one of the drives, so at least one of the drives in the library must be connected to the workstation. This workstation must be an HPSS node running the PVR. The tape driver must be installed on this node. The LTO PVR and SCSI PVR (if using SCSI-3 interface) are used to communicate with these libraries.

3.4.2.3. STK L40, STK SL500, STK SL8500

The SCSI PVR is used to communicate with the STK L40, STK SL500, and STK SL8500.

3.4.2.4. STK

The STK PVR must be able to communicate with STK's ACSLS server. HPSS requires ACSLS version 7.0.0. For the PVR to communicate with the ACSLS server, it must have a TCP/IP connection to the server (e.g. Ethernet) and STK's SSI software must be running on the node with the PVR. Multiple STK Silos can be connected via pass through ports and managed by a single ACSLS server. This collection of robots can be managed by a single HPSS PVR.

3.4.2.5. ADIC AML



The AML PVR is supported by special bid only.

The Distributed AML Server (DAS) client components on the AIX workstations must be able to communicate (via a TCP/IP connected network) with DAS Client components on the node controlling the robot in order to request DAS services. The AML PVR is used to communicate with the ADIC AML.

3.4.3. Manually Mounted Tape

An Operator PVR is used to manage a homogeneous set of manually mounted drives. Tape mount requests will be displayed on an SSM screen.

3.4.4. Tape Devices

The tape devices/drives supported by HPSS are listed in Table 2.



9840C drives should not be used in conjunction with either 9840A or 9840B drives.

Table 2. Supported Platform/Driver/Tape Drive Combinations

Platform	Driver	Device(s)
AIX	IBM	3580 (Gen3, Gen4), 3592 (Gen2, Gen3)
	Native	3580 (Gen3, Gen4), 9840 (C, D), 9940 (A, B), T10000 (A, B)
Linux	Native	3580 (Gen3, Gen4), 3592 (Gen2, Gen3), 9840 (C, D), 9940 (A, B), T10000 (A, B)
IRIX	Native	3580 (Gen3)

The “Driver” column uses the following abbreviations:

IBM IBM SCSI Tape Device Driver (Atape)

Native AIX, IRIX, or Linux native SCSI Tape Device Driver



Older tape drives (3590, 3590E, 3590H, 9840 (A & B), 3580 (Gen1 & Gen2), 3592 (Gen1 & Gen2), DST-312, DST-314) will continue to be supported for existing HPSS sites until they can be upgraded.

3.4.4.1. Multiple Media Support

HPSS supports multiple types of media for certain drives. Listed in the following table is a preference list for each media type that can be mounted on more than one drive type. When the PVL starts, it determines the drive type that each type of media may be mounted on. It makes these decisions by traversing each media type’s list and using the first drive type from the list that it finds configured in the system. For example, referring to Table 2, it can be seen that a single-length 3590E tape will mount on a double-length 3590E drive if and only if there are no single-length 3590E drives configured in the system.

IBM 3592 and 3580 tape technologies support multi-generation reads and writes. The HPSS PVL which is responsible for creating the mount list, does not know the difference between a read and a write request. Thus the PVL will first attempt to mount a particular generation of cartridge into the same generation of drive. If the drives matching the generation of the cartridge are all busy, then it will attempt to mount the cartridge into the next generation drive (if available). For example, if a system has 3580 (LTO) Gen2, 3580 (LTO) Gen3, and 3580 (LTO) Gen4 cartridges and drives, a 3580 (LTO) Gen2 cartridge would be mounted into a 3580 (LTO) Gen2 drive (if not busy). If all the 3580 (LTO) Gen2 drives were busy, it would then attempt to mount the 3580 (LTO) Gen2 cartridge into a 3580 (LTO) Gen3 drive.

Since 3580 LTO tape drives support reading (but not writing) of two previous generations (e.g. 3580

(LTO) Gen4 drive can read 3580 (LTO) Gen4, 3580 (LTO) Gen3, and 3580 (LTO) Gen2 cartridges, but can only write 3580 (LTO) Gen4 and 3580 (LTO) Gen3 cartridges), HPSS will mount a 3580 (LTO) Gen2 cartridge into a 3580 (LTO) Gen4 drive only if 3580 (LTO) Gen2 drives are not defined in HPSS and 3580 (LTO) Gen3 drives are either busy or not defined. In this case, it is up to the HPSS Administrator to make sure these 3580 (LTO) Gen2 cartridges are read only.

Table 3. Cartridge/Drive Affinity Table

Cartridge Type	Drive Preference List
AMPEX DST-312	AMPEX DST-312 AMPEX DST-314
AMPEX DST-314	AMPEX DST-314
Single-Length 3590	Single-Length 3590 Double-Length 3590 Single-Length 3590E Double-Length 3590E Single-Length 3590H Double-Length 3590H
Double-Length 3590	Double-Length 3590 Double-Length 3590E Double-Length 3590H
Single-Length 3590E	Single-Length 3590E Double-Length 3590E Double-Length 3590H
Double-Length 3590E	Double-Length 3590E Double-Length 3590H
Single-Length 3590H	Single-Length 3590H Double-Length 3590H
Double-Length 3590H	Double-Length 3590H
3580 (LTO) Gen 1	3580 (LTO) Gen 1 3580 (LTO) Gen 2 3580 (LTO) Gen 3
3580 (LTO) Gen 2	3580 (LTO) Gen 2 3580 (LTO) Gen 3 3580 (LTO) Gen 4
3580 (LTO) Gen 3	3580 (LTO) Gen 3 3580 (LTO) Gen 4

3580 (LTO) Gen 4	3580 (LTO) Gen 4
3592 J1A Short Tape 3592 J1A Standard Tape	3592 J1A 3592 EO5 3592 EO6
3592 EO5 JJ Short Tape 3592 EO5 JA Standard Tape 3592 EO5 JB XL Tape	3592 EO5 3592 EO6
3592 EO6 JJ Short Tape 3592 EO6 JA Standard Tape 3592 EO6 JB XL Tape	3592 EO6
STK 9840A	STK 9840A STK 9840B STK 9840C STK 9840D
STK 9840B	STK 9840B STK 9840C STK 9840D
STK 9840C	STK 9840C STK 9840D
STK 9840D	STK 9840D
STK 9940A	STK 9940A STK 9940B
STK 9940B	STK 9940B
STK T10000A	STK T10000A STK T10000B
STK T10000B	STK T10000B

Note that the PVL's choices are made at startup time, and are not made on a mount-to-mount basis. Therefore a single-length 3590E cartridge will never mount on a double-length 3590E drive if a single-length 3590E drive was configured in the system when the PVL was started.

3.4.5. Disk Devices

HPSS supports locally-attached disk devices, including those devices attached via SCSI, SSA or

Fibre Channel. For these devices, operating system disk partitions of the desired size must be created (e.g., AIX logical volume or Linux/Solaris/IRIX disk partition), and the raw device name must be used when creating the Mover Device configuration (see Section 7.1: *Configure a New Device & Drive* of the *HPSS Management Guide* for details on configuring storage devices).

3.4.6. Special Bid Considerations

The following options are available by special bid only:

- ADIC AML Tape Libraries
- Sony GY-8240
- HPSS High Availability

3.5. HPSS Sizing Considerations

There are two types of storage space that must be planned for: HPSS User Storage Space & HPSS Infrastructure Storage Space.

HPSS User Storage Space is the disk and tape storage resources that will store user data. Typically, disk storage is allocated at the top level of storage hierarchies and is used as a disk cache for user data. Tape storage is usually allocated to lower levels of storage hierarchies and is used for the long-term, permanent storage of user data.

HPSS Infrastructure Storage Space is the disk space allocated to file systems that contain executables, log files, server metadata (DB2 database), backups, and other HPSS supporting files and data. Tape resources outside of HPSS are usually required for backups of the operating system, HPSS specific file systems, and HPSS metadata unless other backup processes have been configured.

During the HPSS planning phase, it is important to assess how much disk space will be required to support the HPSS production environment. The first step in this process is to understand the various metadata tables managed by the HPSS system. The sections that follow explain the metadata table layout and how to best estimate disk space allocation to support these tables.

How these resources are interconnected to the overall system is just as important as the amount of disk and/or number of tape drives/cartridges allocated. For instance, if there are terabytes of disk storage in the form of several FC disk arrays and 50 enterprise type tape drives, but only one mover and a couple of FC adapters, it is unlikely that the storage system will be able to adequately move data in/out and within the system to meet anyone's demands and expectations. The "data pipes" between the storage resources must be adequately provisioned to allow for efficient transfer of data; including those times of peak demand. In the other extreme, one or both of the storage ends can be under allocated and waste the overall potential of the infrastructure. If there are too few tape drives, data stalls on the disk resources preventing new files from being transferred into the storage system, or from being staged back from tape media in a timely manner when the user requests access to it.

In regard to metadata space, the key consideration is that HPSS is a database application and its performance is directly correlated to how well, or poorly, it uses the database and the storage behind it. Referring to Figure 5, it is probably a waste to allocate 14 36GB drives for HPSS filesystems and the DB2 database. Except for the largest HPSS systems containing hundreds of millions of files, the disk space will never be fully utilized. However, the overriding concern isn't space utilization, it is database transaction performance, which is a function of the efficiency of the disks, controllers, and adapters supporting the database operations. This is the reason that so much attention is focused on the HPSS Core Server and the disk subsystem attached to it. High performance disk and tape systems for user data storage have to be balanced with high performance file systems supporting HPSS and its databases.

Starting with HPSS 6.2 there are many enhancements to the storage system to take advantage of Storage Area Networks. Though separated in Figure 4, in reality there is usually only one SAN at an installation and all the resources are attached to it. Besides the HPSS Movers being connected to SAN, the end-user clients are often SAN attached as well. The result is that the data paths take greater advantage of the SAN architecture and fewer store-and-forward operations are needed through Movers (i.e. clients transfer data across SAN directly to disk resources, the mover just manages the transfer) and less traffic across the TCP/IP network infrastructure. Adequately provisioning the "data pipes" is still critical, but the equation has changed to rely more heavily on the SAN to carry the data traffic.

3.5.1. HPSS User Storage Space

HPSS files are stored on the media that is defined and allocated to HPSS. Enough storage space must be provided to meet the demands of the user environment. HPSS assists in the management of space by providing SSM screens with information about total space and used space in all of the defined storage classes. In addition, alarms can be generated automatically based on configurable threshold values to indicate when space used in a given Storage Class has reached a threshold level. In a hierarchy where data is being migrated from one hierarchy level to a lower one, management of space in the Storage Class provided is done via the migration and purge policies that are provided. The basic factors involved are the total amount of media space available in the Storage Class being migrated and the rate at which this space is used. This will drive how the migration and purge policies are set up for the Storage Class. For more details on this, see Section 3.9.1: *Migration Policy* on page 94 and Section 3.9.2: *Purge Policy* on page 95. Failure to provide enough storage space to satisfy a user request results in the user receiving a NO SPACE error. It is important to understand that the Core Server writes files only to the top level of the COS hierarchy. If the top level does not have sufficient free space for the write operation, it will fail, regardless of the amount of free space in lower levels.

3.5.2. HPSS Infrastructure Storage Space

Figure 5: *Basic HPSS Metadata and Filesystem Allocation* represents what is typically needed to support a basic HPSS server node configuration. Specific function and required space allocation for the defined items will be discussed in following sections. For now, the diagram helps to define the road map between raw, physical disks and the required filesystems and logical volumes/partitions to operate an HPSS system.

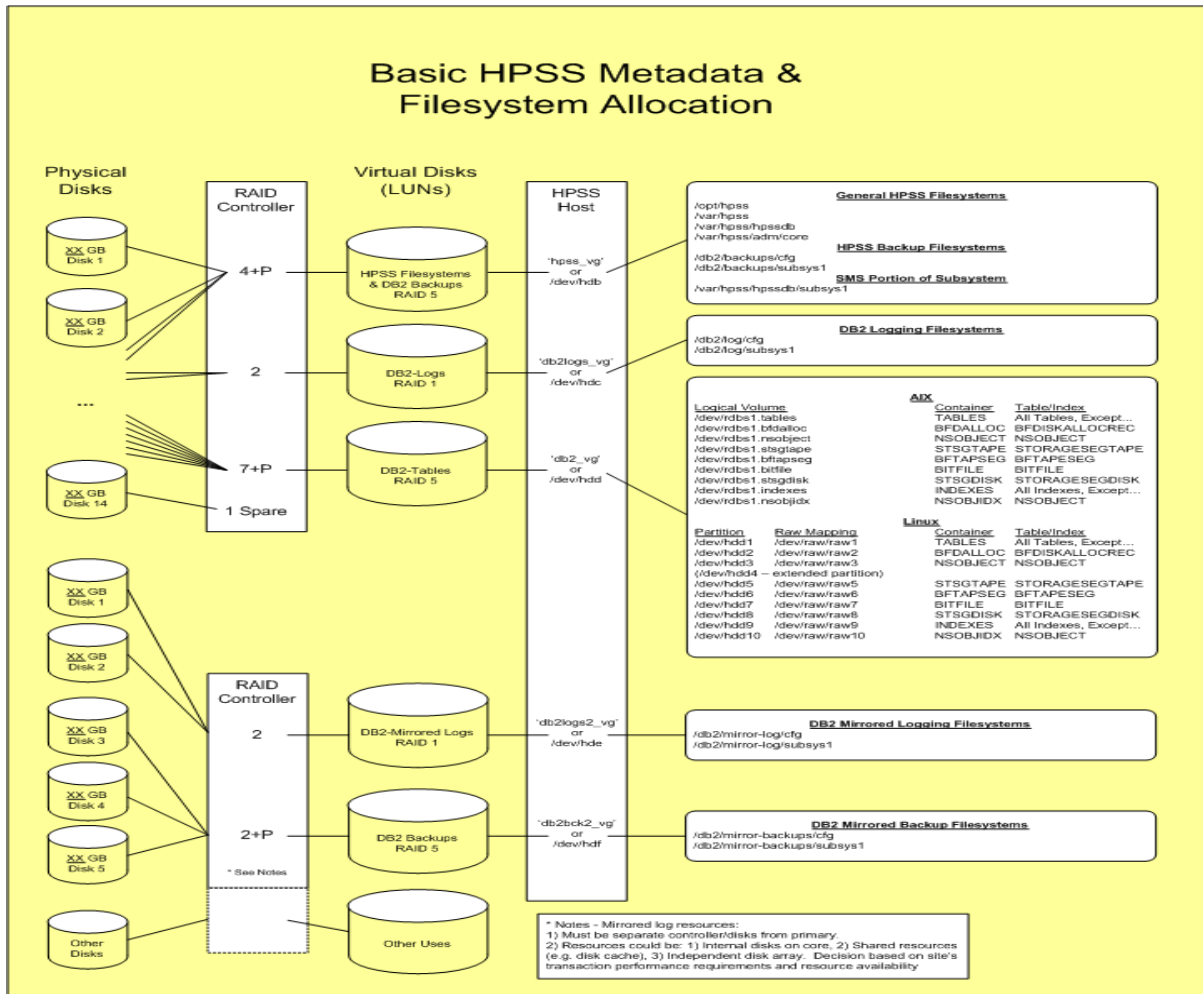


Figure 5. Basic HPSS Metadata & Filesystem Allocation

On the left hand side of the diagram, the raw physical volumes are shown attached to the disk array controller. The configuration of the disks by the controller and its software should be divided into three separate LUNs: 1) HPSS Filesystems and DB2 Backups, 2) DB2 Logs, 3) and the DB2 Tables. One disk may be kept as a "hot spare" in the event that one of the other disks fails. This allows the system to automatically replace the failed media and rebuild the data without immediate intervention from an operator or administrator. Recommended configurations for the LUNs is to use RAID 5, which is good for small, random data access, for the first and third LUN. For the LUN associated with DB2 Logs, it is suggested that RAID 1 (Mirroring) be used since these logs are typically very large and accessed sequentially by the system. Once defined, these LUNs are assigned to the core server host via the LVM (AIX) into a Volume Group or by the operating system (Linux) into a hard drive identifier. The last step is to allocate individual filesystems and logical volumes(AIX)/partitions(Linux) as defined on the right-hand side of the diagram.

HPSS requires the use of DB2 log mirroring. This protects the active DB2 log from possible loss due to the primary metadata disk array failure or other unavailability. To safe guard the DB2 log, separate disk/controller components are required for the LUNs providing storage for the filesystem where the log is written as well as where the archived copies of the logs are stored before being backed up to tape media. It is imperative that the components be totally separate from the primary metadata disk array (i.e. different disks, different controllers, different path such as the FC connections) to provide protection against failures. Specific resource assignment for the the

mirrored log components will need to be determined by HPSS and the customer based on transaction performance requirements. Potentially, disk resources primarily allocated for HPSS disk cache can be used or the site may want to dedicate a second disk array for this purpose to prevent any possible interference.

3.5.3. HPSS Filesystems

The following sections describe the various filesystems used by HPSS.

3.5.3.1. /opt/hpss

The HPSS software is installed in the **/opt/hpss** directory. The installation package sizes and disk requirements are listed in Table 10: *Installation Package Sizes and Disk Requirements*.

3.5.3.2. /var/hpss

See Appendix F: */var/hpss Files* for a more detailed explanation of directories and files located in **/var/hpss**.

The **/var/hpss** directory tree is the default location of a number of HPSS configuration files and other files needed by the servers. It is recommended that this filesystem be at least 1GB in size.

Within the **/var/hpss** filesystem the following sub-directories exist:

- The **/var/hpss/etc** is the default directory where some additional UNIX configuration files are placed. These files are typically very small.
- The **/var/hpss/ftp** is the default directory where several PFTP Daemon files are maintained. There are three sub-directories required: **adm** (contains the **hpss_ftp.log** file), **daemon**, and **daemon/ftpd** where the **ftp.pids-hpss_class** file will be placed.
- The **/var/hpss/tmp** is the default directory where the Startup Daemon creates a lock file for each of the HPSS servers it brought up in the node. HPSS may also write diagnostic log files and disk allocation maps in this directory, when configured to do so. The lock files are very small, but the logs and disk maps may be several tens of kilobytes, or larger.



*It is up to the administrator to remove unneeded reports to prevent the **/var/hpss** filesystem from filling.*

- The **/var/hpss/log** is the default directory where the Log Daemon creates two central log files. The size of these log files is specified in the Log Daemon specific configuration. By default, these files are 5 MB each. In this same directory, the Log Client will continually write a circular ASCII log whose size is defined by the specific configuration. By default, the size of this file is also 5 MB.
- If MPS is configured to produce a migration/purge report, it will generate a report every 24 hours. The size of these reports varies depending on the number of files being migrated and purged during the report cycle. These reports are placed in the **/var/hpss/mps** directory by default and will need to be removed when no longer needed.



*It is up to the administrator to remove unneeded reports to prevent the **/var/hpss** filesystem from filling.*

- If the Gatekeeper is configured to do Gatekeeping Services, then the administrator may wish

to create a site policy configuration file, usually named `/var/hpss/gk/gksitepolicy`. The size of this file depends on the site-implemented gatekeeping policy. If the Gatekeeper Service is not used, there is a minimal amount of disk space used in this directory.

- If an Accounting report is requested, a report file and a checkpoint file are created in the directory specified in the Accounting Policy, usually `/var/hpss/acct`. The sizes of these files depend upon the number of files stored in HPSS.



It is up to the administrator to remove unneeded reports to prevent the `/var/hpss` filesystem from filling.

- If SSM is configured to buffer alarm and event messages in a disk file, a file to store alarms and events will be created in `/var/hpss/ssm` directory. This alarm file is approximately 5 MB in size.

3.5.3.3. `/var/hpss/adm/core`

`/var/hpss/adm/core` is the default directory where HPSS servers put “core” files if they terminate abnormally. Core files may be large, so it is recommended that there should be at least 2 GB reserved for this purpose on the server node and at least 1 GB on Mover nodes.



It is up to the administrator to remove unneeded core files to prevent the `/var/hpss` filesystem from filling.

3.5.3.4. `/var/hpss/hpssdb`

The `/var/hpss/hpssdb` filesystem stores the DB2 instance configuration information and 'CFG' database tables. Specifics on its size are described in the 'CFG' Database Allocation section below. The recommended minimum filesystem size is 1GB.

3.5.3.5. `/var/hpss/hpssdb/subsys1 & subsysX`

The `/var/hpss/hpssdb/subsys1` filesystem stores the configuration and SMS temporary tables allocated for the primary HPSS subsystem database. Specifics on the size of this filesystem are described below in the 'SUBSYS' Database Allocation section. If multiple subsystems are defined in the system, then separate filesystems should be allocated for each. Conventionally, subsystem filesystems are numbered in sequence starting with '1'.

3.5.3.6. `/db2/backups/cfg`

The `/db2/backups/cfg` filesystem temporarily stores backup images of the CFG log archives and database as they are generated. The backup files are then transferred to long-term media, such as tape, using a backup file manager such as TSM. Details are described in the DB2 Disk Space section. The recommended minimum filesystem size is 2GB.



Ensure that it has sufficient space to store the data generated by each DB2 backup; otherwise the backup will fail. If the required backup space exceeds the maximum size for a JFS filesystem (approximately 63.8 GB), consider using a JFS2 filesystem. Other options include having DB2 compress the backup data as it is taken or having DB2 store the backup data onto more than one filesystems.

3.5.3.7. /db2/backups/subsys1 & subsysX

Similar to `/db2/backups/cfg`, the `/db2/backups/subsys1` filesystem temporarily stores backup images of the subsystem archived logs and database as they are generated. The backup files are then transferred to long-term media, such as tape, using a backup file manager such as TSM. Details are described in the Section 3.5.4.4: *DB2 Disk Space* on page 77. Separate filesystems are allocated for each subsystem defined in the system.



Ensure that it has sufficient space to store the data generated by each backup; otherwise the backup will fail. If the required backup space exceeds the maximum size for a JFS filesystem (about 63.8 GB), consider using a JFS2 filesystem. Other options include having DB2 compress the backup data as it is taken or having DB2 store the backup data onto more than one filesystems.

3.5.3.8. /db2/log/cfg

Though small in comparison to the DB2 subsystem logging area, the `/db2/log/cfg` filesystem is recommended to store the 'CFG' transaction logs in a separate filesystem from the configuration and 'CFG' tables.

3.5.3.9. /db2/log/subsys1 & subsysX

The `subsys1` and subsequent subsystem databases require a separate filesystem to store the DB2 transaction logs. The `/db2/log/subsys1` filesystem size is discussed in Section 3.5.4.4: *DB2 Disk Space* on page 77.

3.5.3.10. /db2/mirror-log/cfg

This filesystem is similar to that of `/db2/log/cfg`, but the storage resources for the DB2 log must be separate from the resource of the primary copy. DB2 is configured so that it writes to both the primary and secondary log copies as transactions are registered by the database.

3.5.3.11. /db2/mirror-log/subsys1 & subsysX

This filesystem is similar to that of `/db2/log/subsys1`, but the storage resources for the DB2 log must be separate from the resource of the primary copy. DB2 is configured so that it writes to both the primary and secondary log copies as transactions are registered by the database.

3.5.3.12. /db2/mirror-backup/cfg

This filesystem is smaller than `/db2/backups/cfg` since it will only contain archived copies of the mirrored copy of the 'CFG' logs. Similar to `/db2/backups/cfg`, it needs to be managed by a backup tool such as TSM.

3.5.3.13. /db2/mirror-backup/subsys1 & subsysX

This filesystem is smaller than `/db2/backups/subsys1` since it will only contain archived copies of the mirrored copy of the 'SUBSYS1' logs. Similar to `/db2/backups/subsys1`, it needs to be managed by a backup tool such as TSM.

3.5.3.14. SUBSYS1 Database Allocation

The recommended allocation of disk space for the SUBSYS1 (and other subsystems) database is shown in Figure 5: on page . By default, **mkhpss** will distribute the database tables and indexes across the 9 logical volumes/partitions shown in the diagram. Specifics of **mkhpss** are described in

Section 5.3.1.2: *Install HPSS Documentation and DB2 Software* on page 141. The tables and indexes are separated into separate logical volumes/partitions to ease future expansion of the database and to maximize performance of database operations.

For Linux, access to a /dev/hdxy partition is through the Linux buffered I/O system. While this is an appropriate access method for a filesystem that supports journaled logging, for DB2 and Mover accesses, non-buffered IO is required. Linux, up to release RHEL 4.0, has provided 'raw device' access mode through the use of the 'raw' command and interface. Before DB2 uses the partitions defined in the Figures 5 through Figure 8, the mapping of the /dev/hdxy device and the raw interface must be configured by the administrator.

Though RHEL 4.0 and later supports the LVM manager, HPSS configurations have not attempted to use logical volumes created on Linux by LVM for DB2 storage and their use is not recommended at this time.

3.5.4. HPSS Metadata Space

During the HPSS planning phase, it is important to properly assess how much disk space will be required by DB2 to store HPSS metadata. The first step in this process is to understand the metadata tables managed by DB2. The sections that follow explain the metadata table layout and how best to estimate disk space allocation to support these tables.

3.5.4.1. SMS versus DMS Space

DB2 table spaces can be allocated either as System Managed Space (SMS) or Database Managed Space (DMS). The SMS allocation method uses a local filesystem which is managed by the operating system. DB2 creates files in the filesystem to store the tables and indexes. In the DMS allocation method, DB2 manages raw disk space directly, bypassing the operating system's buffers and filesystem interface. We recommend the use of SMS for storing short or seldom used tables, and DMS for storing large tables frequently used tables.

Tables used to define the configuration and policies of the system are typically small. These are contained in the configuration or global database, usually named 'CFG', and should be allocated in SMS space. Tables such as the BITFILE or STORAGESEGTAPE tables can be quite large and their placement must be optimized for performance. These tables are contained in one or more subsystem databases, usually called 'SUBSYSx' (where x is the subsystem number), and should be allocated in DMS space.

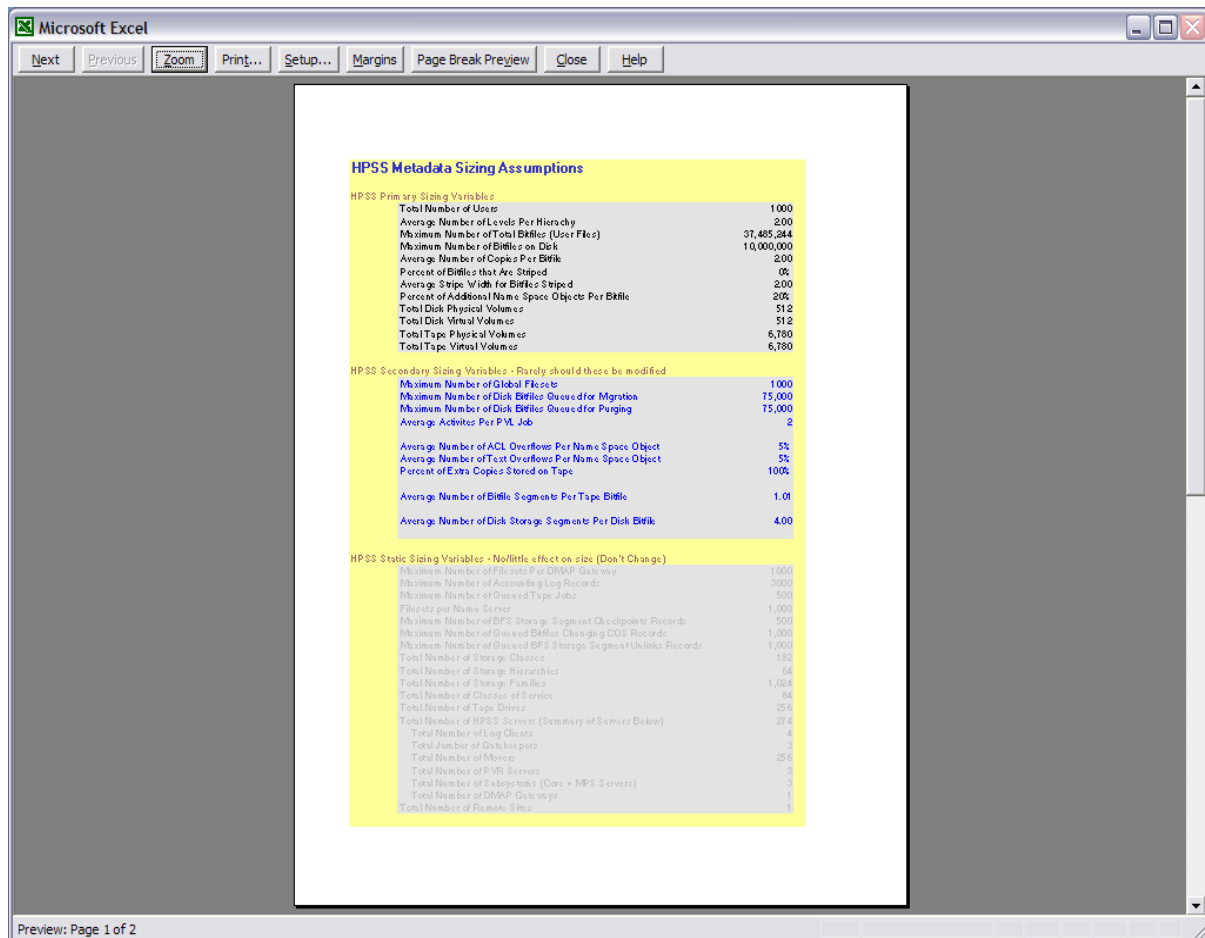
3.5.4.2. 'CFG' Database Allocation

By default, **mkhps** will store the DB2 related files for HPSS in the **/var/hpss/hpssdb** directory. As recommended above, this directory should be a separate filesystem of RAID disks. The amount of space needed will vary somewhat depending upon the number of subsystems in use. For a site with only one subsystem, the amount of space should be about 1GB. For each additional subsystem, an additional 1/2GB should be allocated.

3.5.4.3. 'SUBSYS' Database Allocation

HPSS support representatives will provide sites with the "6.2 Sizing Spreadsheet" to help determine the amount of disk resources required to store the HPSS metadata and what the allocation should be across the DB2 tablespaces. The total amount of disk space should be allocated as a 'raw logical volume' on AIX or allocated as a raw device on Linux. The space should be allocated on a RAID or mirrored disk device. The total space can span several logical devices as necessary. DB2 space can be allocated using **mkhps** which provides the option to define the space or use existing equally sized

disk devices. The “6.2 Sizing Spreadsheet” input tab is shown below.



Based on the input, the resulting output is show below:

Microsoft Excel

Next Previous Zoom Print... Setup... Margins Page Break Preview Close Help

Tables/page	Page size (B)	# of Pages	Size (Bytes)	Size (GBs)	SaMh Factor	# of Pages (w/ Safety Factor)	Size (Bytes)-w/ SaMh Factor	Size (GBs)-w/ SaMh Factor	Actual used pages	Actual SaMh factor
HUBSYS										
BYCATSPACE	4	7,468	30,670,848	0.03	1.1	8,237	33,727,853	0.03	7,468	
TEUPSPACE1	4	3,282,322	13,798,937,716	14.71	1.1	4,241,969	17,375,089,287	16.18		
BFDALOC	4	1,333,930	5,462,733,230	5.09	1.1	1,467,043	6,009,003,930	5.60	88,912	0.1
BFTAPESEG	4	2,448,199	10,019,390,936	9.33	1.1	2,689,872	11,016,286,364	10.26	1,289,664	0.8
BITFILE	4	2,777,282	11,378,749,998	10.68	1.1	3,068,010	12,613,301,883	11.69	2,932,032	1.1
INDEXES	4	4,978,912	20,382,108,931	19.38	1.6	7,463,419	30,570,924,346	28.47	2,189,732	0.4
NSOBJECT	4	2,210,842	13,297,370,829	12.48	1.1	2,397,507	14,727,197,882	13.73	3,100,034	0.9
NSOBJV	4	1,038,482	4,649,420,311	4.18	0.8	2,751,204	11,149,500,777	10.39	1,346,504	1.1
BFSODSK	4	2,091,760	8,404,008,980	7.83	1.1	2,286,939	9,244,479,389	8.61	89,024	0.0
BFTAPE	4	3,398,332	13,798,937,716	14.71	1.1	4,241,969	17,375,089,287	16.18	2,035,248	0.8
TABLES	4	948,262	3,871,793,736	3.61	1.1	1,039,703	4,263,973,177	3.97	394,190	0.4
Total		22,744,409	108,997,004,739	101.60		32,783,171	134,278,367,100	126.08	13,236,704	0.6
INDEXES										
BYCATSPACE	4	15,711	64,352,236	0.06	1.1	17,282	70,787,482	0.07		
TEUPSPACE1	8	268	1,081,429	0.00	1.1	249	2,036,872	0.00		
USERSPACE1	8	363	709,429	0.01	1.1	1,021	4,384,888	0.01		
Total		18,866	71,897,644	0.07		18,662	81,199,758	0.08		

Actual Row Count	Table	Estimated row count	Rows/Page	Pages	Size (Bytes)	Total Table size including indexes	% of total Table space	
HUBSYS								
	AB BADDR	74,970,463	109.07	686,129,911	2,810,388,096	7,374,197,960	7.92% TABLES	
	ABEADDR-IDX1			148.22	816,246,78	2,114,646,812	INDEXES	
	ABEADDR-IDX2			128.26	897,864,811	2,449,203,052	INDEXES	
	ACOTLOG	3,000	76.53	39,06	159,914	282,767	0.00% TABLES	
	ACOTLOG-IDX1			119.80	25,10	102,828	INDEXES	
	ACOTENAP	2,000	83.94	23,91	97,844	229,016	0.00% TABLES	
	ACOTENAP-IDX1			82.50	32,00	131,072	INDEXES	
	ACOTBUM	2,000	86.00	26,36	148,847	280,019	0.00% TABLES	
	ACOTBUH-IDX1			82.50	32,00	131,072	INDEXES	
	BFDISKHANDLE	1,000	89.16	16,78	89,789	219,210	0.00% TABLES	
	BFDISKHANDLE-IDX1			82.50	16,00	68,636	INDEXES	
	BFDISKHANDLE-IDX2			48.25	20,72	84,888	INDEXES	
	BFDISKALLOCREC	10,000,000	7.45	1,339,800,000	5,462,733,230	6,016,204,900	6.40% BFDALOC	
	BFDISKALLOCREC-IDX1			74.01	136,120,000	583,481,620	INDEXES	
	BFDISKREQ	10,000,000	89.49	188,100,000	688,637,600	1,288,893,800	1.38% TABLES	
	BFDISKREQ-IDX1			80.49	148,000,000	590,160,000	INDEXES	
	BFDISKREQ-IDX2			80.48	1,240,50	5,081,038	16,977,101	0.02% TABLES
	BFDISKREQ-IDX3			82.50	1,200,000	4,915,200	INDEXES	
	BFDISKREQ-IDX4			44.01	1,704,30	6,800,313	INDEXES	
	BFDISKREQ-IDX5			71.90	1,047,45	4,290,356	12,368,913	0.01% TABLES
	BFDISKREQ-IDX6			72.16	1,039,05	4,256,949	INDEXES	
	BFDISKREQ-IDX7			79.39	307,65	1,240,514	INDEXES	
	BFTAPECHKPT	500	44.09	11,34	48,449	167,667	0.00% TABLES	
	BFTAPECHKPT-IDX1			48.25	10,36	42,443	INDEXES	
	BFTAPECHKPT-IDX2			20.00	19,23	73,766	INDEXES	
	BFTAPECHKPT-IDX3			46.00	22,22	91,030	266,524	0.00% TABLES
	BFTAPECHKPT-IDX4			24.75	40,40	169,499	INDEXES	
	BFTAPE	75,720,193	33.97	2,446,168,617	10,019,390,936	14,549,782,900	16.62% BFTAPESEG	
	BFTAPE-IDX1			88.40	1,107,029,222	4,634,391,636	INDEXES	
	BITFILE	37,488,244	13.80	2,777,281,73	11,378,749,998	13,110,742,979	14.07% BITFILE	
	BITFILE-IDX1			88.50	423,823,24	1,734,897,521	INDEXES	
	DMGFILESET	1,000	15.46	64,71	269,036	386,103	0.00% TABLES	

Preview: Page 1 of 4

Definitions for the DB2 tables are as follows:

Bitfile Disk Allocation Maps. (BFDISKALLOCREC) For each bitfile stored on disk, one or more rows will be created in the Disk Allocation Maps table. The number of rows is determined by the storage segment size in the Storage Class in which the files are stored and the average file size stored in that Storage Class.

Bitfile Disk Segments. (BFDISKSEG) For a bitfile stored on disk, one or more rows will be created in the bitfile disk segment table. Because bitfile disk segments track the contiguous pieces of a bitfile, there normally will be only one disk segment row per file.

Bitfile Tape Segments. (BFTAPESEG) For a bitfile stored on tape, one or more rows will be created in the bitfile tape segment table. Under normal conditions, one bitfile tape segment row is expected for each tape on which the file is stored. It is safe to assume for each bitfile stored on tape, two bitfile segment records are needed.

Bitfiles. (BITFILE) One row is created in the bitfile metadata table for each bitfile. The amount of space allocated for this DB2 table will normally limit how many bitfiles can be created. Regardless of how many copies of a bitfile exist and whether the bitfile is spread across disk and/or tape, only one bitfile row is created for the file.

Name Space Objects. (NSOBJECT) Each name space object (file, fileset, directory, hard link, junction or soft link) uses one row in the NSOBJECT table.

Name Space Text. (NSTEXT) A name space text row exists for each name space object that has a comment or is a symbolic link. These text rows are variable length with a maximum size of 1023

bytes.

Disk Storage Segments. (STORAGESEGDISK) Expect the size of the disk storage segment metadata table to be quite volatile. As files are added to HPSS, disk storage segments will be created, and as files are migrated to tape and purged from disk, they will be deleted. A rough estimate of the number of disk storage segments can be obtained by estimating the number of files that will be resident on disk in the subsystem for each Storage Class, then multiplying by the Average Number of Segments parameter of the Storage Class.

Tape Storage Segments. (STORAGESEGTAPE) The tape storage segment metadata table grows as the number of files stored in the subsystem increases. Storage segments are created to describe contiguous chunks of data written on tapes. Segment sizes may vary greatly. The number of storage segments found on a given tape is not limited by the Core Server. This number is a function of the length of the files written on the tape, the VV block size, the size of the data buffer used to write the tape, and other factors.

HPSS support representatives will help sites use this spreadsheet as they size the metadata disk resources and allocate those resources for the various DB2 tablespaces.

3.5.4.4. DB2 Disk Space

This section explains the local file system disk space requirements to support DB2, other than for database tables.

The disk space that DB2 requires falls into the following categories:

DB2 log



The DB2 log requires a filesystem in which transaction log files are stored. The number of log files and size of the files is controlled by several database configuration parameters. LOGFILSIZ determines the size of each log file and LOGPRIMARY + LOGSECOND determine the total number of log files that could exist in the logging filesystem that you specify with NEWLOGPATH. Each database should log to its own filesystem and must utilize a RAID device for storing the data. Also, it is required to use the DB2 MIRRORLOGPATH configuration variable and define a separate filesystem to store the mirrored log file.

When DB2 is first started, log files are placed in the `/var/hpss/<instance_name>/NODE0000/...` directory. After DB2 is started, use the database configuration parameters NEWLOGPATH and MIRRORLOGPATH to direct log files to a prepared location.

To determine the amount of disk space required for the DB2 log files, run:

```
% db2 get db cfg for <db name>
```

Look for the LOGFILSIZ variable, which is the size of each log file, and multiply it by the total number of logs for this database (LOGPRIMARY + LOGSECOND). The result will be the number of 4KB blocks needed. To convert this to the number of bytes, multiply the result by 4096. Calculate this for each database that utilizes this particular log filesystem to determine the total necessary disk space.

The optimal type of disk used for the log filesystem(s) is a low-latency disk since the I/O is performed in many small chunks. To ensure full recoverability in the event of media failure on the log disk, it is important to mirror the log on a separate physical disk.

DB2 database home directories

DB2 requires approximately 700MB of disk space for installation. Upon installation each database

created will require approximately 10MB of disk space in a filesystem that should be used only for database home directories and other DB2 home directories (such as the DB2 Administration Server and Fenced User). This filesystem should be protected by RAID since DB2 may require information in the database home directory to properly recover a database.

DB2 backups

Full backups of DB2 databases may be larger than the databases themselves. The amount of local disk space needed for backups will depend largely on the backup strategy.

When using the DB2 to TSM strategy, local disk is not needed to backup DB2 databases. DB2 communicates directly with the TSM API to deliver backup images and archived log files directly to TSM. This strategy is highly recommended.

3.5.5. System Memory and Disk Space

The following sections discuss recommendations and requirements for disk space, system memory, and paging space.

3.5.5.1. Operating System Disk Spaces

It is recommended that all operating system logical volumes/partitions be mirrored. This is true of the HPSS server and Mover nodes. Both AIX and Linux support this type of configuration.

3.5.5.2. System Disk Space Requirements for Running SSM

The SSM Graphical User Interface, **hpssgui**, and Command Line Interface, **hpssadm**, have an option to create session log files. **hpssgui** records all status bar and popup messages issued during the session in its log. **hpssadm** records all prompts, error and informational messages, and requested output (lists, managed objects, configuration structures, etc.) issued during the session in its log. Old session log files should be removed periodically to avoid filling the file system. This is typically done with a **cron** job. For example, the following command will remove all files from **/tmp** which have not been accessed within the previous seven days:

```
% find /tmp -atime +7 -exec rm {} \;
```

The creation of the session log files is controlled by the **-S** option to the **hpssgui** and **hpssadm** startup scripts. See their man pages for details.

3.5.5.3. System Memory and Paging Space Requirements

The memory and disk space requirements for the nodes where the HPSS Servers will execute depends on the configuration of the servers, the nodes that each server will run on, and the amount of concurrent access they are configured to handle.

At least 2GB of memory is recommended for nodes that will run one or more HPSS servers (and most likely a DB2 server), excluding the HPSS Movers. More memory is required for systems that run most of the servers on one node and/or support many concurrent users. The memory available to HPSS and DB2 servers is critical to providing acceptable response times to end user operations. Disk space requirements are primarily covered by Section 3.5.4: *HPSS Metadata Space* on page 74 for DB2 space, and the preceding subsections under Section 3.5.5: *System Memory and Disk Space* on page 78 for the individual HPSS servers. Sufficient disk space should be allocated for the paging space, using recommendations in the system documentation for the amount of memory configured.

The amount of memory for nodes running HPSS Movers, and no DB2 servers, is dependent on the number and types of devices configured on the Mover node, the expected usages of those devices, and the configuration of the Movers. In general, Movers supporting disk devices will require more memory than Movers supporting tape devices because disk devices are likely to have more outstanding requests. At least 1GB of memory should be configured on the Mover nodes. More memory is required for nodes that support many devices, especially disks, and have large numbers of concurrent end-user requests. Additionally, the size of the Mover's internal buffers impacts the Mover's memory requirements. Two buffers are used by each Mover process to handle I/O requests.

Paging space should be sized according to the following rules:

Table 4. Paging Space Info

Amount of physical memory	Minimum recommended amount of paging space
memory <= 256MB	2 * amount of physical memory
256MB < memory <= 1GB	512MB + ((amount of physical memory – 256MB) * 1.25)
1GB < memory <= 2GB	1.5 * amount of physical memory
2GB < memory	1 * amount of physical memory

3.6. HPSS Interface Considerations

This section describes the user interfaces to HPSS and the various considerations that may impact the use and operation of HPSS.

3.6.1. Client API

The HPSS Client API provides a set of routines that allow clients to access the functions offered by HPSS. The API consists of a set of calls that are comparable to the file input/output interfaces defined by the POSIX standard (specifically ISO/IEC 9945-1:1990 or IEEE Standard 1003.1-1990), as well as extensions provided to allow access to the extended capabilities offered by HPSS.

The Client API is built on top of the HPSS security layer (either UNIX or Kerberos). It must be run on a platform that supports the Core Server's security layer. For example if the Core Server is using Kerberos authentication then users on the client platform must be able to authenticate themselves with the Core Server's Kerberos realm. To access HPSS from client platforms that do not support the Core Server's security layer, FTP or Parallel FTP must be used.

The Client API allows clients to specify the amount of data to be transferred with each request. The amount requested can have a considerable impact on system performance and the amount of metadata generated when writing directly to a tape storage class. See Section 3.9.6: *Location Policy* on page 99 and Section 3.11: *HPSS Performance Considerations* on page 112 for further information.

The details of the Application Program Interface are described in the *HPSS Programmer's Reference Guide*.

3.6.2. FTP

HPSS provides an FTP daemon that supports standard FTP clients. Extensions are also provided to allow additional features of HPSS to be utilized and queried. Extensions are provided for specifying Class of Service to be used for newly created files, as well as directory listing options to display

Class of Service and Accounting Code information. In addition, the **chgrp**, **chmod**, and **chown** commands are supported as **quote site** options.

The FTP daemon is built on top of the Client API and must be run on a node that supports Kerberos clients. Note that FTP clients can run on computers that do not have Kerberos installed.

The size of the buffer, used for reading and writing HPSS files, can be specified in the FTP daemon configuration. The buffer size selected can have a considerable impact on both system performance and the amount of metadata generated when writing directly to a tape Storage Class. See Section 3.9.6: *Location Policy* on page 99 and Section 3.11: *HPSS Performance Considerations* on page 112 for further information.

The GSSFTP from MIT is supported if the HPSS FTP Daemon is appropriately configured. This client provides credential-based authentication and “Cross Realm” authentication to enhance security and “password-less” FTP features.

Refer to the *HPSS User's Guide* for details of the FTP interface.

3.6.3. Parallel FTP

The FTP daemon also supports the HPSS Parallel FTP (PFTP) protocol, which allows the PFTP client to utilize the HPSS parallel data transfer mechanisms. This provides the capability for the client to transfer data directly to the HPSS Movers (i.e., bypassing the FTP Daemon), as well as the capability to stripe data across multiple client data ports (and potentially client nodes). Data transfers are supported through TCP/IP. Support is also provided for performing partial file transfers.

The FTP protocol is supported by the HPSS FTP Daemon. Refer to Section 13.2: *FTP Daemon Configuration* of the *HPSS Management Guide* for configuration information. No additional configuration of the FTP Daemon is required to support PFTP clients.

The client side executable for PFTP is **pftp_client**. **pftp_client** supports TCP based transfers. Because the client executable is a superset of standard FTP, standard FTP requests can be issued as well as the PFTP extensions. Authentication using either username/password or Kerberos credentials is configurable.

Refer to the *HPSS User's Guide* for details of the PFTP interface.

3.6.4. XFS

XFS for Linux is an open source filesystem from SGI based on SGI's XFS filesystem for IRIX.

The impression of an infinitely large XFS filesystem can be achieved by using HPSS as a back end to XFS. HPSS transparently archives inactive XFS data into HPSS storage which frees up XFS disk space for active data.

It is well suited to sites with large numbers of small files or clients who wish to use NFS to access HPSS data. However, the files can only be accessed through XFS (or NFS via XFS) and cannot be accessed with HPSS utilities such as parallel FTP.

3.7. HPSS Server Considerations

Servers are the internal components of HPSS that provide the system's functionality. They must be configured correctly to ensure that HPSS operates properly. This section outlines key considerations that should be kept in mind when planning the server configuration for an HPSS system.

3.7.1. Core Server

The Core Server is responsible for managing the HPSS name space (files, directories, links, etc.), bitfiles, and storage (physical volumes, virtual volumes, etc.) for a single subsystem. Each of these areas of responsibility are outlined in greater detail below.

Core Server at large

The Core Server uses POSIX threads to service concurrent requests. The Core Server accepts requests from any authenticated client; however, certain Core Server functions can be performed only by trusted clients. Trusted clients are those for whom control permission has been set in the Core Server's ACL entry for the client. Higher levels of trust are granted to clients who have both control and write permission set in their ACL entry. Refer to Section 2.1: *HPSS Server Security ACLs* of the *HPSS Management Guide* for information concerning the ACL for the Core Server.

The Core Server can be configured to allow or disallow super-user privileges (root access). When the Core Server is configured to allow root access, the UID of the super-user is configurable.

HPSS systems configured with multiple subsystems employ multiple Core Servers and multiple metadata databases. Though the servers are separate, each Core Server in a given HPSS realm must share the fileset global metadata table.

Name Space

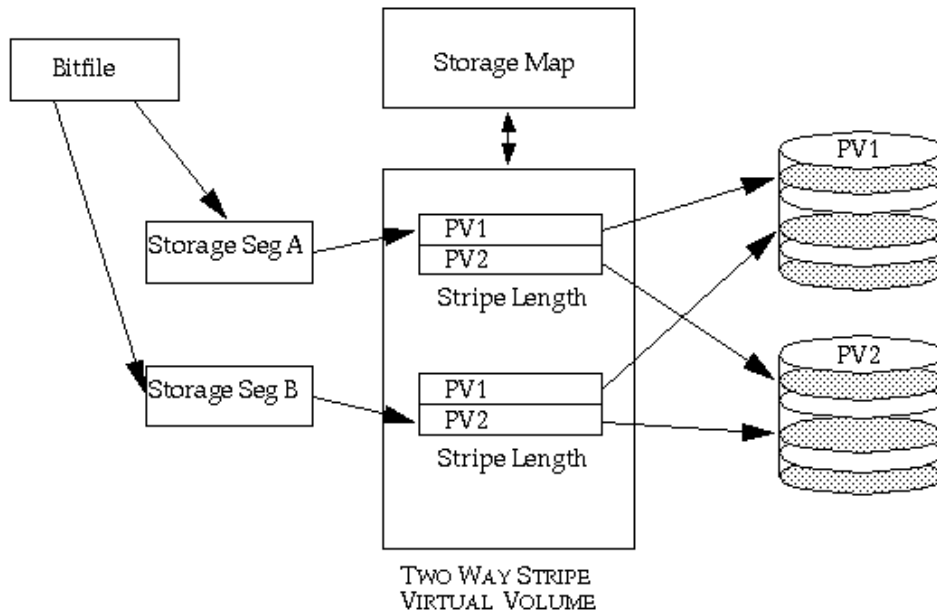
The HPSS Core Server maintains the HPSS name space in system metadata. Refer to Section 3.5: *HPSS Sizing Considerations* on page 68 for details on sizing the name space. Refer to Section 14.8: *DB2 Space Shortage* of the *HPSS Management Guide* for information on handling a metadata space shortage. By utilizing multiple storage subsystems, it is possible to distribute large name spaces across multiple Core Servers. Junctions between the names spaces of these storage subsystems can be used to "join" these subsystems.

Bitfiles

The Core Server provides a view of HPSS as a collection of files. It provides access to these files and maps the files onto underlying storage objects.

When a Core Server is configured, it is assigned a server ID. This value should never be changed because it is embedded in the ID of each bitfile and storage segments it uses. HPSS expects to be able to extract the Server ID from any bitfile ID and connect to that server to access the file.

The Core Server maps bitfiles to their underlying physical storage by maintaining information that maps a bitfile to the storage segments that contain its data. For additional information, see Section 3.7.1: *Core Server* on page 81. The relationship of bitfiles to storage segments and other structures is shown in Figure 9: *The Relationship of Various Server Data Structures*.



RELATIONSHIP BETWEEN BITFILES, SEGMENTS, MAPS, VV AND PV
 Figure 6. The Relationship of Various Server Data Structures

Disk Storage Allocation

Each Core Server manages disk storage units for HPSS. It maps each disk storage unit onto an HPSS disk Physical Volume (PV) and records configuration data for the PV. Groups of one or more PVs (disk stripe groups) are managed by the server as disk Virtual Volumes (VVs). The server also maintains a storage map for each VV that describes which portions of the VV are in use and which are free. Figure 9: *The Relationship of Various Server Data Structures* shows the relationship of Core Server data structures such as VVs to other server data structures.

The server can manage information for any number of disk PVs and VVs; however, because a copy of all of the PV, VV, and storage map information is kept in memory while the server runs, the size of the server will be somewhat proportional to the number of disks it manages.

The Core Server is designed to scale up its ability to manage disks as the number of disks increase. As long as sufficient memory and CPU capacity exist, threads can be added to the server to increase its throughput. Additional subsystems can also be added to a system, increasing concurrency even further.

Tape Storage Allocation

Each Core Server manages serial access magnetic tape storage media for HPSS. The server maps each tape volume onto an HPSS tape PV and records configuration data for the PV. Groups of one or more PVs (tape stripe groups) are managed by the server as tape VVs. The server maintains a storage map for each VV that describes how much of each tape VV has been written and which storage segment, if any, is currently writable in the VV. Figure 9: *The Relationship of Various Server Data Structures* shows the relationship of data structures such as VVs to other server data structures.

The server can manage information for any number of tape PVs and VVs. It can also manage an unlimited number of tape PVs, VVs, maps, and segments without impacting its memory size.

The Core Server is designed to scale up its ability to manage tapes as the number of tapes increases. As long as sufficient memory and CPU capacity exist, threads can be added to the server to increase its throughput. Additional subsystems can also be added to a system, increasing concurrency even further.

Note that the number of tape drives the server manages has much more to do with the throughput of the server than the number of tape volumes the server manages. If the number of tape drives in the system needs to increase to meet workload demands, adding a new subsystem and redistributing the drives may be the best way to deal with the increased workload.

3.7.2. Migration/Purge Server

The Migration/Purge Server (MPS) reports storage class usage statistics and manages the amount of free space available in storage classes by performing periodic migration and purge runs on the storage classes. Migration copies data from the storage class on which it runs to one or more lower levels in the storage hierarchy. Once data has been migrated, a subsequent purge run will delete the data from the migrated storage class, if so configured. Migration is a prerequisite for purge, and MPS will never purge data which has not previously been migrated. It is possible, but not desirable, to assign only a migration policy and no purge policy to a disk storage class; however, this will result in data being copied (migrated) but never deleted (purged). It is important to recognize that migration and purge policies determine when data is migrated from a storage class and when the data is purged from that storage class; however, the number of copies and the location of those copies is determined by the storage hierarchy definition. The MPS uses the Core Server to both perform data movement between hierarchy levels and gather statistics. As such, the Core Server must be running in order for the MPS to complete its duties.

The MPS can only exist within a storage subsystem and a subsystem may be configured with no more than one MPS. Storage hierarchies are global across all storage subsystems within an HPSS system, but a given hierarchy may or may not be enabled within a given subsystem (a hierarchy is enabled within a subsystem by configuring the subsystem to enable one or more classes of service which reference that hierarchy). Note that a storage class may not be selectively migrated and purged in different subsystems. If the hierarchy contains storage classes which require migration and purge, then an MPS must be configured to run against those storage classes in the subsystem. This MPS will manage migration and purge operations on only those storage resources within its assigned subsystem. Thus, for an HPSS system with multiple storage subsystems, there may be multiple MPSs, each operating on the resources within a particular subsystem.

Migration and purge operate differently on disk and tape storage classes. Disk migration and disk purge are configured on a disk storage class by associating a migration policy and a purge policy with that storage class. For tape storage classes, the migration and purge operations are combined, and are collectively referred to as tape migration. Tape migration is enabled by associating a migration policy with a tape storage class. Purge policies are not needed or supported on tape storage classes.

Once migration and purge policies are configured for a storage class (and the MPS is restarted), the MPS will begin scheduling migration and purge runs for that storage class. Migration on both disk and tape is run periodically according to the runtime interval configured in the migration policy. Disk purge runs are not scheduled periodically, but rather are started when the percentage of space used in the storage class reaches the threshold configured in the purge policy for that storage class. It is critical that the hierarchies to which a storage class belongs be configured with proper migration targets in order for migration and purge to perform as expected.

The purpose of disk purge is to maintain a given amount of free space in a disk storage class by removing data of which copies exist at lower levels in the hierarchy. The order in which purge records are sorted, which determines the order in which files are purged, may be configured on the purge policy. It should be noted that all of the options except *Record Create Time* require additional

metadata updates and can impose extra overhead on DB2. Also, unpredictable purge behavior may be observed if the purge record ordering is changed with existing purge records in the system until these existing records are cleared. A purge run ends when either the supply of purge records is exhausted or the purge target is reached.

There are two different tape migration methods, tape volume migration and tape file migration. The method which is applied to a tape storage class is selected in the migration policy for that storage class. Tape volume migration's goal is freeing tape volumes by moving data segments from sparsely filled volumes either laterally (to another tape within the same storage class) or vertically (to a lower level in the storage hierarchy). Tape file migration can be thought of as a hybrid between the disk and tape volume migration methods. It is a file-based tape method which is able to make a single copy of tape files to the immediately lower level in the hierarchy. For details on tape migration options, please see Section 3.9.1.2: *Migration Policy* on page 95.

If, at any point during a tape file migration run, the MPS detects that the source tape volume has become active, migration is abandoned on this volume until the next migration run. This is done in order to avoid competing with an HPSS system user for this volume. A tape volume is deemed to be active if any file it contains has been read or written within the access intervals specified in the migration policy.

The MPS provides the capability of generating migration/purge report files that document the activities of the server. The specification of the UNIX report file name prefix in the MPS server specific configuration enables the server to create these report files. It is suggested that a complete path be provided as part of this file name prefix. Once reporting is enabled, a new report file is started every 24 hours. The names of the report files are made up of the UNIX file name prefix from the server specific configuration, plus a year-month-day suffix. With reporting enabled, MPS will generate file-level migration and purge report entries in real time. These report files can be interpreted and viewed using the `mps_reporter` utility. Since the number and size of the report files grow rapidly, each site should develop a cron job that will periodically remove the reports that are no longer needed.

In order to efficiently perform disk migration, the MPS parallelizes the migration of files from disk to tape. The number of files that the MPS migrates simultaneously is user configurable via the Request Count in the *Disk Migration Policy*. For example, if the Request Count is set to one, then the MPS will serially migrate files. If the Request Count is set to four, then the MPS will attempt to have four files migrating at a time.

As previously indicated, the MPS provides the information displayed in the *HPSS Active Storage Classes* window in SSM. Each MPS contributes storage class usage information for the resources within its storage subsystem. MPS accomplishes this by polling the Core Server within its subsystem at the interval specified in the MPS server specific configuration. The resulting output is one line for each storage class for each storage subsystem in which that class is enabled. The MPS for a subsystem does not report on storage classes which are not enabled within that subsystem. The warning and critical storage class thresholds are also activated by the MPS.

3.7.3. Gatekeeper

Each Gatekeeper may provide sites with the ability to:

- Schedule the use of HPSS resources using Gatekeeping Services.
- Validate user accounts using the Account Validation Service.

If the site doesn't want either service, then it is not necessary to configure a Gatekeeper into the HPSS system.

Sites can choose to configure zero (0) or more Gatekeepers per HPSS system. Gatekeepers are associated with storage subsystems. Each storage subsystem can have zero or one Gatekeeper associated with it and each Gatekeeper can support one or more storage subsystems. Gatekeepers are associated with storage subsystems using the *Storage Subsystem Configuration* screen (see Section 4.2: *Storage Subsystems* of the *HPSS Management Guide*). If a storage subsystem has no Gatekeeper, then the Gatekeeper field will be blank. A single Gatekeeper can be associated with every storage subsystem, a group of storage subsystems, or one storage subsystem. A storage subsystem can NOT use more than one Gatekeeper.

Every Gatekeeper has the ability to supply the Account Validation Services. A bypass flag in the Accounting Policy metadata indicates whether or not Account Validation for an HPSS system is on or off. Each Gatekeeper will read the Accounting Policy metadata file, so if multiple Gatekeepers are configured and Account Validation has been turned on, then any Gatekeeper can be chosen by the Location Server to fulfill Account Validation requests.

Every Gatekeeper has the ability to supply the Gatekeeping Service. The Gatekeeping Service provides a mechanism for HPSS to communicate information through a well-defined interface to a policy software module to be completely written by the site. The site policy code is placed in a well-defined site shared library for the gatekeeping policy (`/opt/hpss/lib/libgksite.[a|so]`) which is linked to the Gatekeeper. The gatekeeping policy shared library contains a default policy which does NO gatekeeping. Sites will need to enhance this library to implement local policy rules if they wish to monitor and/or load balance requests.

The gatekeeping site policy code will determine which types of requests it wants to monitor (authorized caller, create, open, and stage). Upon initialization, each Core Server will look for a Gatekeeper in the storage subsystem metadata. If no Gatekeeper is configured for a particular storage subsystem, then the Core Server in that storage subsystem will not attempt to connect to any Gatekeeper. If a Gatekeeper is configured for the storage subsystem that the Core Server is configured for, then the Core Server will query the Gatekeeper asking for the monitor types by calling a particular Gatekeeping Service API. This API will then call the appropriate Site Interface which each site can provide to determine which types of requests are to be monitored. This query by the Core Server will occur each time the Core Server (re)connects to the Gatekeeper. The Core Server will need to (re)connect to the Gatekeeper whenever the Core Server or Gatekeeper is restarted. Thus if a site wants to change the types of requests it is monitoring, then it will need to restart the Gatekeeper and Core Server.

If multiple Gatekeepers are configured for gatekeeping, then the Core Server that controls the files being monitored will contact the Gatekeeper that is located in the same storage subsystem. Conversely if one Gatekeeper is configured for gatekeeping for all storage subsystems, then each Core Server will contact the same Gatekeeper.

A Gatekeeper registers five different interfaces: Gatekeeper Services, Account Validation Services, Administrative Services, Connection Manager Services, and Real Time Monitoring Services. When the Gatekeeper initializes, it registers each separate interface. The Gatekeeper specific configuration will contain any pertinent data about each interface.

The Gatekeeper Service interface provides the Gatekeeping APIs which calls the site implemented Site Interfaces. The Account Validation Service interface provides the Account Validation APIs. The Administrative Service provides the server APIs used by SSM for viewing, monitoring, and setting server attributes. The Connection Manager Service provides the HPSS connection management interfaces. The Real Time Monitoring Service interface provides the Real Time Monitoring APIs.

The Gatekeeper Service Site Interfaces provide a site the mechanism to create local policy on how to throttle or deny create, open and stage requests and which of these request types to monitor. For example, it might limit the number of files a user has opened at one time; or it might deny all create

requests from a particular host or user. The Site Interfaces will be located in a shared library that is linked into the Gatekeeper.

It is important that the Site Interfaces return a status in a timely fashion. Create, open, and stage requests from MPS are timing sensitive, thus the Site Interfaces won't be permitted to delay or deny these requests, however the Site Interfaces may choose to be involved in keeping statistics on these requests by monitoring requests from Authorized Callers.


If a Gatekeeper should become heavily loaded, additional Gatekeepers can be configured (maximum of one Gatekeeper per storage subsystem). In order to keep the Gatekeepers simple and fast, they do not share state information. Thus if a site wrote a policy to allow each host a maximum of 20 creates, then that host would be allowed to create 20 files on each storage subsystem that has a separate Gatekeeper.

The Gatekeeper's Real Time Monitoring Interface supports clients such as a Real Time Monitoring utility which requests information about particular user files or HPSS Request Ids.

3.7.4. Location Server

All HPSS client API applications, which includes all end user applications, will need to contact the Location Server at least once during initialization and usually later during execution in order to locate the appropriate servers to contact. If the Location Server is down for an extended length of time, these applications will eventually give up retrying their requests and become non-operational. To avoid letting the Location Server become a single point of failure, consider replicating it, preferably on a different node. If replicating the Location Server is not an option or desirable, consider increasing the automatic restart count for failed servers in SSM. Since the Location Server's requests are short lived, and each client contacts it through a cache, performance alone is not usually a reason to replicate the Location Server. Generally the only time a Location Server should be replicated solely for performance reasons is if it is reporting heavy load conditions to SSM.

If any server is down for an extended length of time it is important to mark the server as non-executable within SSM. As long as a server is marked executable the Location Server continues to advertise its location to clients which may try to contact it.

 *The Location Server must be reinitialized or recycled whenever the Location Policy or its server configuration is modified. Note that it is not necessary to recycle the Location Server if an HPSS server's configuration is added, modified, or removed since this information is periodically reread.*

3.7.5. PVL

The PVL is responsible for mounting and dismounting PVs (such as tape and magnetic disk) and queuing mount requests when required drives and media are in use. The PVL usually receives requests from Core Server clients. The PVL accomplishes any physical movement of media that might be necessary by making requests to the appropriate Physical Volume Repository (PVR). The PVL communicates directly with HPSS Movers in order to verify media labels.

The PVL is not required to be co-resident with any other HPSS servers and is not a CPU-intensive server. With its primary duties being queuing, managing requests, and association of physical volumes with PVRs, the PVL should not add appreciable load to the system.

In the current HPSS release, only one PVL will be supported.

3.7.6. PVR

The PVR manages a set of imported cartridges, mounts and dismounts them when requested by the

PVL. It is possible for multiple HPSS PVRs to manage a single robot. This is done if it is necessary to organize the tape drives in the robot into partitions. Each tape drive in the robot is assigned to exactly one PVR. Additionally, each cartridge is assigned to only one PVR. The PVRs can be configured identically and can communicate with the robot through the same interface.

The following sections describe the considerations for the various types of PVRs supported by HPSS.

3.7.6.1. STK PVR

The STK PVR communicates to the ACSLS server via STK's SSI software.

The SSI must be started before the PVR. If the SSI is started after the PVR, the PVR should be stopped and restarted.

If multiple STK robots are managed, SSIs that communicates with each of the robots should be configured on separate CPUs. A PVR can be configured on each of the CPUs that is running an SSI. If multiple STK robots are connected and are controlled by a single Library Management Unit (LMU), a single PVR can manage the collection of robots. The PVR can be configured on any CPU that is running an SSI.

HPSS is tested with Storage Technology Corporation's (STK's) Automated Cartridge System Library Software (ACSL) Version 7.0.0.

ACSL should be running on the workstation directly connected to the STK Silo. The HPSS STK PVR can run on any workstation that has a TCP/IP connection to the ACSLS workstation. The workstation running the HPSS PVR must also be running STK's Storage Server Interface (SSI) software. This software will not be started by HPSS and should be running when HPSS is started. It is recommended that the SSI be started by the workstation's initialization scripts every time the workstation is booted.

The SSI requires that the system environment variables `CSI_HOSTNAME` and `ACSAPI_PACKET_VERSION` be correctly set. Note that due to limitations in the STK Developer's Toolkit, if the SSI is not running when the HPSS PVR is started, or if the SSI crashes while the HPSS PVR is running, the HPSS PVR will lock up and will have to be manually terminated by issuing "**kill -9 <pid>**".

3.7.6.2. LTO PVR

The LTO PVR manages the IBM 3584 Tape Library and Robot, which mounts, dismounts and manages LTO tape cartridges and IBM 3580 tape drives. The PVR uses the Atape driver interface to issue SCSI commands to the library.

The SCSI control path to the library controller device (`/dev/smc*`) is shared with the first drive in the library (typically `/dev/rmt0`). Since the PVR communicates directly to the library via the Atape interface, the PVR must be installed on the same node that is attached to the library.

The LTO PVR operates synchronously; that is, once a request is made to the 3584 library, the request thread does not regain control until the operation has completed or terminated. This means that other requests must wait on an operation to complete before the PVR can issue them to the 3584.

HPSS is designed to work with the AIX tape driver (Atape) software to talk to the IBM 3584 LTO Library over a SCSI channel. Currently HPSS is only supported for the AIX version of the Atape driver. Please note that the PVR must run on the same node that has the Atape interface and this node must have a direct SCSI connection to the library.

Please refer to The 3584 UltraScalable Tape Library Planning and Operator Guide and IBM Ultrium Device Drivers Installation and User's Guide for more information.

3.7.6.3. 3494 PVR

The 3494 PVR can manage an IBM 3494 tape robot attached via Ethernet or SCSI. The PVR will create a process to receive asynchronous notifications from the robot.

At least one PVR should be created for every robot managed by HPSS. If multiple 3494 robots are managed, care must be taken to ensure that the PVRs are configured to communicate with the correct `/dev/lmcp` devices. The PVRs can run on the same CPU or different CPUs as long as the proper `/dev/lmcp` devices are available.

HPSS is designed to work with IBM 3494 robots attached to an HPSS server with either RS-232 or Ethernet control connections. Data paths to the drives will be SCSI-2 with RS-232 and Ethernet control paths. The HPSS PVR must run on a machine with the appropriate version of Library Manager Control Point (LMCP) device drivers installed.

3.7.6.4. AML PVR



The AML PVR is supported by special bid only.

The AML PVR can manage ADIC AML robots that use Distributed AML Server (DAS) software. The DAS AML Client Interface (ACI) operates synchronously; that is, once a request is made to the AML, the request process does not regain control until the operation has completed or terminated. Therefore, the AML PVR must create a process for each service request sent to the DAS (such as mount, dismount, eject a tape, etc.).

HPSS is designed to work with ADIC Distributed Automated Media Library Server (DAS) software version 1.3 and the ABBA Management Unit (AMU) version 2.4.0. DAS is the ADIC software which consists of the Automated Media Library (AML) Client Interface (ACI) and the DAS server components. The AMU is the host computer software by which the ADIC Storage System manages the archive database, which is based on a DB2 compatible database for an OS/2 system.

The AMU must run on a OS/2 PC host computer connected to the AML robot while the HPSS AML PVR can run on any RS/6000 workstation that has a TCP/IP connection to the OS/2 host computer. The workstation running the HPSS AML PVR must also contain the DAS/ACI software that is called by the HPSS AML PVR.

Refer to ADIC DAS Installation and Administration Guide and Reference Guide AMU for additional information.

3.7.6.5. Operator PVR

The Operator PVR displays mount requests for manually mounted drives. The mount requests are displayed on the appropriate SSM screen

All of the drives in a single Operator PVR must be of the same type. Multiple operator PVRs can be configured without any additional considerations.

3.7.6.6. SCSI PVR

The SCSI PVR communicates with tape libraries and robots through a generic SCSI interface. The interface uses the SCSI-3 command set. The SCSI PVR currently supports the following medium changers: IBM 3584, IBM Ultrium 3582, STK L40, STK SL500, STK SL8500, and Spectralogic.

3.7.7. Mover

The Mover configuration is largely dictated by the hardware configuration of the HPSS system. Each Mover can handle both disk and tape devices and must run on the node to which the storage devices are attached. The Mover is also capable of supporting multiple data transfer mechanisms for sending data to or receiving data from HPSS clients (e.g., TCP/IP and shared memory).

3.7.7.1. AIX Asynchronous I/O

Asynchronous I/O must be enabled manually on AIX platforms. There should be no asynchronous I/O setup required for Solaris, Linux, or IRIX platforms.

To enable asynchronous I/O on an AIX platform, use either the **chdev** command:

```
% chdev -l aio0 -a autoconfig=available
```

or **smitty**:

```
% smitty aio
<select "Change / Show Characteristics of Asynchronous I/O">
<change "STATE to be configured at system restart" to "available">
<enter>
```

Asynchronous I/O on AIX must be enabled on the nodes on which the Mover will be running.

3.7.7.2. Tape Devices

All tape devices that will be used to read and write HPSS user data must be set to handle variable block sizes to allow for the ANSI standard 80-byte volume label and file section headers. This section describes the procedure for setting this option on each supported operating system.

3.7.7.2.1. AIX

To set the devices to use variable blocks on an AIX platform, either use the **chdev** command (substituting the appropriate device name for **rmt0**):

```
% chdev -l rmt0 -a block_size=0
```

or **smitty**:

```
% smitty tape
<select "Change / Show Characteristics of a Tape Drive">
<select the appropriate tape device>
<change "BLOCK size (0=variable length)" to "0">
<enter>
```

Note: Take into account differences in the interface based on the specific device driver supporting the device

3.7.7.2.2. Solaris

For Solaris, the method used to enable variable block sizes for a tape device is dependent on the type of driver used. Supported devices include Solaris SCSI Tape Driver and IBM SCSI Tape Driver.

For the IBM SCSI Tape Driver, set the **block_size** parameter in the **/opt/IBMtape/IBMtape.conf** configuration file to **0** and perform a reboot with the reconfiguration option. The Solaris SCSI Tape

Driver has a built-in configuration table for all HPSS supported tape drives. This configuration provides variable block size for most HPSS supported drives. In order to override the built-in configuration, device information can be supplied in the `/dev/kernel/st.conf` as global properties that apply to each node.

Consult the tape device driver documentation for instructions on installation and configuration.

3.7.7.2.3. IRIX

Variable block sizes can be enabled for the IRIX native tape device driver by configuring the Mover to use the tape device special file with a “v” in the name (e.g. `/dev/rmt/tps5d5nsvc`).

3.7.7.2.4. Linux

HPSS supports tape devices on Linux with the use of the native SCSI tape device driver (**st**). To enable the loading of the Linux native tape device, uncomment the following lines in the `".config"` file and follow the procedure for rebuilding your Linux kernel.

```
CONFIG_SCSI=y
CONFIG_CHR_DEV_ST=y
```

In Linux, tape device files are dynamically mapped to SCSI IDs/LUNs on your SCSI bus. The mapping allocates devices consecutively for each LUN of each device on each SCSI bus found at the time of the SCSI scan, beginning at the lower LUNs/IDs/buses. The tape device file will be in this format: `/dev/st[0-31]`. This will be the device name to use when configuring the HPSS device.

3.7.7.3. Disk Devices

All locally attached magnetic disk devices (e.g., SCSI, SSA) should be configured using the pathname of the raw device (i.e., character special file).

For Linux systems, this may involve special consideration.

HPSS supports disk device on Linux with the use of the native SCSI disk device driver (**sd**) and the raw device driver (**raw**).

The Linux SCSI Disk Driver presents disk devices to the user as device files with the following naming convention: `/dev/sd[a-h][0-8]`. The first variable is a letter denoting the physical drive, and the second is a number denoting the partition on that physical drive. Occasionally, the partition number will be left off when the device corresponds to the whole drive. Drives can be partitioned using the Linux **fdisk** utility.

The Linux raw device driver is used to bind a Linux raw character device to a block device. Any block device may be used.

See the Linux manual page for more information on the SCSI Disk Driver, the Raw Device Driver and the **fdisk** utility.

To enable the loading of the Linux native SCSI disk device, uncomment the following lines in the configuration file and follow the procedure for rebuilding your Linux kernel.

```
CONFIG_SCSI=y
CONFIG_BLK_DEV_SD=y
```

Also, depending on the type of SCSI host bus adapter (HBA) that will be used, you will need to enable one or more of the lower level SCSI drivers. For example, if you are using one of the Adaptec

HBAs with a 7000 series chip set, uncomment the following lines in the ".config" file and follow the procedure for rebuilding your Linux kernel.

```
CONFIG_SCSI_AIC7XXX=y
CONFIG_AIC7XXX_CMDS_PER_DEVICE=253
CONFIG_AIC7XXX_RESET_DELAY_MS=15000
```

3.7.7.4. Performance

The configuration of the Movers and attached devices can have a large impact on the performance of HPSS because of constraints imposed by a number of factors e.g., device channel bandwidth, network bandwidth, processor power.

A number of conditions can influence the number of Movers configured and the specific configuration of those Movers:

- Each Mover process is built to handle a specific device interface, e.g., IBM SCSI-attached 3590/3590H/3580 drives. If multiple types of devices are to be supported, multiple Movers must be configured.
- Each Mover currently limits the number of concurrently outstanding connections. If a large number of concurrent requests are anticipated on the drives planned for a single Mover, the device work load should be split across multiple Movers. This is primarily an issue for Movers that will support disk devices.
- The planned device allocation should be examined to verify that the device allocated to a single node will not overload that node's resource to the point that the full transfer rates of the device cannot be achieved (based on the anticipated storage system usage). To off-load a single node, some number of the devices and their corresponding Mover can be allocated to other nodes.
- In general, the connectivity between the nodes on which the Movers will run and the nodes on which the clients will run should have an impact on the planned Mover configuration. For TCP/IP data transfers, the only functional requirement is that routes exist between the clients and Movers; however, the existing routes and network types will be important to the performance of client I/O operations.
- Mover to Mover data transfers, performed for migration, staging, and repack operations, also impact the Mover configuration. For devices that support storage classes involved in migration or staging, the Movers controlling those devices should be configured so that there is an efficient data path among them. If Movers involved in a data transfer are configured on the same node, the transfer will occur via a shared memory segment.

3.7.8. Logging Service

Logging Services are comprised of the Log Daemon, Log Client, and Delog processes.

If central logging is enabled (the default), log messages from all HPSS servers will be written by the Log Daemon to a common log file. There is a single Log Daemon process per HPSS system.

The Delog process is executed as a command line utility. The central log file must be accessible from the node on which the command is being executed. Refer to Section 9.5.2: *Viewing the Central Log* of the *HPSS Management Guide* for detailed information on Delog.

Implementation of delog services via the SSM GUI is no longer supported. The command line utility, hpsd_delog, must be used.

3.7.9. Startup Daemon

The Startup Daemon is responsible for starting, monitoring, and stopping the HPSS servers. The Daemon responds only to requests from the SSM System Manager. It shares responsibility with each HPSS server for ensuring that only one copy of the server runs at a given time. It helps the SSM determine whether servers are still running, and it allows the SSM to send signals to servers. Normally, the SSM stops servers by communicating directly with them but, in special cases, the SSM can instruct the Startup Daemon to send a **SIGKILL** signal to cause the server to shut down immediately.

If a server is configured to be restarted automatically, the Startup Daemon will restart the server when it terminates abnormally. The Daemon can be configured to restart the server without limit, or up to a fixed number of restarts, or not at all.

Choose a descriptive name for the Daemon that includes the name of the computer where the Daemon will be running. For example, if the Daemon will be running on a computer named *tardis*, use the descriptive name "Startup Daemon (**tardis**)".

The Startup Daemon is started by running the `/opt/hpss/bin/rc.hpss` script. It ignores most signals and may only be killed using the "kill -9 <pid>" command. The Startup Daemon must be run under the **root** account so that it has sufficient privileges to start the HPSS servers.

The Startup Daemon runs on every HPSS server node.

3.7.10. Storage System Management

SSM has three components:

1. System Manager - Communicates with all other HPSS components requiring monitoring or control.
2. GUI (**hpssgui**) - Provides the HPSS administrator or operator the ability to configure or monitor the HPSS System through a set of windows.
3. Command Line Interface (**hpssadm**) - Provides the HPSS administrator or operator the ability to configure or monitor a subset of the HPSS system through a set of interactive or batch commands.

There can be only one SSM System Manager configured for an HPSS installation. The System Manager is able to handle multiple SSM GUI or Command Line clients (on different hosts or on the same host).

Starting up SSM GUI (**hpssgui**) directly from the HPSS server node where SSM System Manager is running and displaying the SSM window on the user's desktop is discouraged. This is due to known Java/X performance problems. Instead, it is recommended to install the Java and HPSS GUI client software on the user's desktop and execute it there. See Section 3.3.5: *SSM Desktop Client Packaging* of the *HPSS Management Guide* for more information.

There are no performance problems associated with running the SSM Command Line Interface (**hpssadm**) directly on the server UNIX platform, and this is the recommended configuration.

Both the SSM GUI client, **hpssgui**, and the SSM Command Line client, **hpssadm**, may be executed on any platform that complies with Section 3.3.2.3.1: *SSM Client Requirements* on page 61.

Before starting the SM, a review of SM key environment variable settings would be wise. Following is a table of key SM environment variables along with the default value and meaning. Depending on the size (number of servers and number of SSM clients) and activity of the HPSS system, these

values may need to be overridden in env.conf.

Key SM Environment Variables

<i>Variable</i>	<i>Default Value</i>	<i>Functionality</i>
HPSS_SM_SRV_CONNECT_FAIL_COUNT	3	Connection Fail Count: number of connection failures to a server before the Max Connection Interval takes affect (*)
HPSS_SM_SRV_CONNECT_INTERVAL_MIN	20	Interval between attempting server connections when Connection Fail Count has not yet been reached (seconds) (*)
HPSS_SM_SRV_CONNECT_INTERVAL_MAX	60	Max Connection Interval: interval between server connections when Connection Fail Count has been reached without a successful connection (seconds) (*)
HPSS_SM_SRV_MONITOR_THREADS	5	Number of threads created to monitor server connections
HPSS_SM_SRV_QUEUE_SIZE	5	Request Queue Size used by the System Manager server interface - default of 5 slots in the server interface request queue to be used when the server interface thread pool is completely full. The queue is used to hold RPC requests from servers until a thread is available to process the request. Note that if the request queue has any entries in it, it means that all the threads in the server thread pool are busy and the SM response will be degraded. If this happens then it would be good to increase the number of threads available to the server interface using the HPSS_SM_SRV_TPOOL_SIZE variable. Increasing the size of the queue will not help with performance.
HPSS_SM_SRV_TPOOL_SIZE	100	Thread Pool Size used by the System Manager server interface. If the Thread Pool is exhausted then server RPC requests will be queued in the server RPC Request Queue to wait for a thread to become available. When the thread pool is exhausted SM performance may be degraded. Increase this value if that is the case. Typically 1 thread per HPSS server should be adequate. But a few extra wouldn't hurt.

<i>Variable</i>	<i>Default Value</i>	<i>Functionality</i>
HPSS_SM_SRV_MAX_CONNECTIONS	50	Number of HPSS server connections to maintain at once. If this number of connections is exceeded, then old connections will be closed to maintain this number of connections

* The SM attempts to throttle the connection attempts to other servers. It will attempt to reconnect to each server every HPSS_SM_SRV_CONNECT_INTERVAL_MIN seconds until the number of failures for that server has reached HPSS_SM_SRV_CONNECT_FAIL_COUNT. After the failure count has been reached the SM will only try to reconnect to the server every HPSS_SM_SRV_CONNECT_INTERVAL_MAX seconds until a successful connection is made at which time the connection interval for the server will be set back to HPSS_SM_SRV_CONNECT_INTERVAL_MIN.

3.8. Storage Subsystem Considerations

Storage subsystems are provided in HPSS for the purpose of increasing the scalability of the system, particularly with respect to the Core Servers. An HPSS system consists of one or more subsystems, and each subsystem contains its own Core Server. If multiple Core Servers are desired, this is accomplished by configuring multiple subsystems.

Each subsystem uses a separate DB2 database. Adding a subsystem to an HPSS system means adding an additional database that must be maintained and backed up.

3.9. Storage Policy Considerations

This section describes the various policies that control the operation of the HPSS system.

3.9.1. Migration Policy

The migration policy provides the capability for HPSS to copy (migrate) data from one level in a hierarchy to one or more lower levels. The migration policy defines the amount of data and the conditions under which it is migrated; however, the number of copies and the location of those copies is determined by the storage hierarchy definition. The site administrator will need to monitor the usage of the storage classes being migrated and adjust both the migration and purge policies to obtain the desired results.

3.9.1.1. Migration Policy for Disk

Disk migration in HPSS copies (migrates) files from a disk storage class to one or more lower levels in the storage hierarchy. Removing or purging of the files from disk storage class is controlled by the purge policy. The migration and purge policies work in conjunction to maintain sufficient storage space in the disk storage class.

When data is copied from the disk, the copied data will be marked purgeable but will not be deleted. Data is deleted by running purge on the storage class. If duplicate copies are created, the copied data is not marked purgeable until all copies have been successfully created. The migration policy and purge policy associated with a disk storage class must be set up to provide sufficient free space to deal with demand for storage. This involves setting the parameters in the migration policy to migrate a sufficient number of files and setting the purge policy to reclaim enough of this disk space to provide the free space desired for users of the disk storage class.

Disk migration is controlled by several parameters. By default, these parameters are the same across all subsystems. However, subsystem-specific policies may be created which override all of these values. For a list of these parameters, refer to Section 6.4.2.2: *Disk Migration Policy Configuration* in the *HPSS Management Guide*.

3.9.1.2. Migration Policy for Tape

There are two tape migration algorithms: tape file migration and tape volume migration. The algorithm which MPS applies to a given tape storage class is selected in the migration policy for that storage class.

The purpose of tape file migration is to make a second copy of files written on tape. This algorithm is similar to disk migration, but only a single additional copy is possible. It is also possible to configure tape file migration such that files are moved downwards in the hierarchy without keeping a second copy.

It is possible for tape file migration to make a second copy of files written on tape. The algorithm is similar to disk file migration, but only a single additional copy is possible. It is also possible to configure tape file migration such that files are moved downwards in the hierarchy without keeping a second copy.

The purpose of tape volume migration is to empty tape virtual volumes that have become full (reached EOM) and have significant unused space in them. Unused space on a tape volume is generated when files on that tape are deleted or overwritten. Since data can only be recorded on tapes sequentially, vacated recording space on tapes can be reclaimed only by moving all remaining files to other tapes.

Tape volume migration attempts to empty tapes by moving data off of the tapes to other volumes. When a tape becomes empty, it is a candidate for reuse. The **reclaim** utility program resets the state of the empty tape volumes so that they can be reused. The **reclaim** utility can be run from SSM, but it should generally be set up to run on a periodic basis via the **cron** facility. For more information on **reclaim**, see Section 8.1.5: *Reclaiming HPSS Tape Virtual Volumes* of the *HPSS Management Guide* and the **reclaim** manual page.

The **repack** utility can also be used to create empty tapes in a storage class. The administrator should determine whether a tape should be repacked based on the number of holes (due to file overwrite or deletion) on the tape. If a tape storage class is at the bottom of a hierarchy, **repack** and **reclaim** must be run periodically to reclaim wasted space. For more information on **repack**, see Section 8.1.4: *Repacking Tape Virtual Volumes* of the *HPSS Management Guide* and the **repack** manual page.

The migration policy parameters which apply to the different tape migration algorithms in detail in Section 6.4.2.3: *Tape Migration Policy Configuration* in the *HPSS Management Guide*.

3.9.2. Purge Policy

The purge policy allows the MPS to remove the bitfiles from disk after the bitfiles have been migrated to a lower level of storage in the hierarchy. A purge policy cannot be defined for a tape storage class or a disk storage class which does not support migration. Sites may or may not wish to define a purge policy for all disk storage classes that support migration. Purging from tapes is controlled by the "Migrate Files and Purge" flag of the tape migration policy; there is no separate purge policy for tape storage classes.

The specification of the purge policy in the storage class configuration enables the MPS to do the disk purging according to the purge policy for that particular storage class. Purge is run for a storage class on a demand basis. The MPS maintains current information on total space and free space in a

storage class by periodically extracting this information from the HPSS Core Server. Based upon parameters in the purge policy, a purge run will be started when appropriate. The administrator can also force the start of a purge run via SSM.

The disk purge is controlled by several parameters:

- The **Do not purge files accessed within <nnn> minutes** parameter determines the minimum amount of time a site wants to keep a file on disk. Files that have been accessed within this time interval are not candidates for purge.
- The **Start purge when space used reaches <nnn> percent** parameter allows the amount of free space that is maintained in a disk storage class to be controlled. A purge run will be started for this storage class when the total space used in this class exceeds this value.
- The **Stop purge when space used falls to <nnn> percent** parameter allows the amount of free space that is maintained in a disk storage class to be controlled. The purge run will attempt to create this amount of free space. Once this target is reached, the purge run will end.
- The **Purge Locks expire after <nnn> minutes** parameter allows the length of time a file can be “purge locked” before it will appear on the MPS report to be controlled. The “purge lock” is used to prevent a file from being purged from the highest level of a hierarchy. Purge locks only apply to a hierarchy containing a disk on the highest level. HPSS will not automatically unlock locked files after they expire. HPSS reports the fact that they have expired in the MPS report.
- The **Purge by** list box allows sites to choose the criteria used in selecting files for purge. By default, files are selected for purge based on their migration time. Alternately, the selection of files for purging may be based on the time the file was created or the time the file was last accessed. Files may be purged in an unpredictable order if this parameter is changed while there are existing purge records already in metadata until those existing files are processed.

Administrators should experiment to determine the parameter settings that will fit the needs of their site. If a site has a large amount of disk file write activity, the administrator may want to have more free space and more frequent purge runs. However, if a site has a large amount of file read activity, the administrator may want to have smaller disk free space and less frequent purge runs, and allow files to stay on disk for a longer time.

3.9.3. Accounting Policy and Validation

The purpose of the Accounting Policy is to describe how a site will charge for storage, and, in addition, to describe the level of user authorization (validation) to be performed when maintaining accounting information.

A site must decide which style of accounting to use before creating any HPSS files or directories. There are two styles of accounting: UNIX-style accounting and Site-style accounting. In addition a site may decide to customize their style of accounting by writing an accounting site policy module for the Gatekeeper.

If a site chooses Site-style accounting and Account Validation is turned off, LDAP must be used as the authorization mechanism. The `hpssGECOS` field which is maintained for each user in LDAP, contains the account index which allows Site-style accounting to be used. However if Account Validation is turned on then the account index comes from the Account Validation metadata (through the Gatekeeper).

If UNIX authorization is used and Account Validation is turned off, UNIX-style accounting must be used because there is no `hpssGECOS` field. The basic limitation is that if the account index is needed

out of the hpssGECOS field, it does not exist in UNIX. It only exists in LDAP.

The metadata for each file and directory in an HPSS system contains an Account field, which determines how the storage will be charged. Each user has at least one default account index, which is put into the Account field of all new files and directories .

When using UNIX-style accounting, the account index is the user's UID. When the user's UID is combined with the user's Realm Id, a unique Account is created.

When using Site-style accounting, each user may have more than one account index, and may switch among them at runtime.

Each site must decide whether it wishes to validate Accounts. However, when using UNIX-style accounting no authorization checking need be done since the account is always the user's UID.

If Account Validation is enabled, additional authorization checks are performed when the following events occur: when files and directories are created, when their ownership is changed, when their account index is changed, or when a user attempts to use an account index other than their default. If the authorization check fails, the operation fails with a permission error.

Using Account Validation is highly recommended for sites that will be accessing remote HPSS systems. The use of Account Validation will help keep account indexes consistent. If remote sites are not being accessed, Account Validation is still recommended as a mechanism to keep consistent accounting information.

If UNIX-style accounting is used, at least one Gatekeeper must be configured .

For Site-style accounting, an Account Validation metadata file must be created, populated and maintained with valid user account indexes. See the Account Validation Editor (hpss_avaledit) manual page for details on the use of the Account Validation Editor.

If the **Require Default Account** field is enabled when using Site-style accounting and Account Validation, users are required to have valid default account indexes before performing almost any client API action. If the **Require Default Account** field is disabled (which is the default behavior) users will only be required to have a valid account set when performing an operation which requires an account to be validated such as a create, an account change operation, or an ownership change operation.

When using Site-style accounting with Account Validation, if the **Account Inheritance** field is enabled, newly created files and directories will automatically inherit their account index from their parent directory. The account indexes can then be changed explicitly by users. This is useful when individual users have not had default accounts set up for them or if entire directory trees need to be charged to the same account. When **Account Inheritance** is disabled (which is the default) newly created files and directories will obtain their account from the user's current session account, which is initially set to the user's default account index. This default account index may be changed by the user during the session.

A site may decide to customize the way they do accounting. In most cases these sites should enable Account Validation with Site-style accounting and then implement their own site policy module which will be linked with the Gatekeeper. See Section 3.7.3: *Gatekeeper* on page 84 as well as the appropriate sections of the *HPSS Programmers Reference* for more information.

By default Account Validation is disabled (bypassed). If it is disabled, the style of accounting is determined by looking up each user's hpssGECOS account information in the authorization registry. The following instructions describe how to set up users in this case.

If a users have their default account index encoded in a string of the form **AA=<default-acct-idx>** in

the principal's LDAP hpssGECOS attribute, then Site-style accounting will be used. Otherwise UNIX-style accounting will be used.

To keep the accounting information consistent, it is important to set up all users in the HPSS Authorization services with the same style of accounting (i.e. they should all have the **AA=** string in their hpssGECOS attribute or none should have this string.) The `hpss_ldap_admin` tool can be used to set attributes for a user including the hpssGECOS field. For more information, see the `hpss_ldap_admin` man page.

See Section 12.4: *Accounting* of the *HPSS Management Guide* for more information.

3.9.4. Security Policy

HPSS server authentication and authorization make extensive use of UNIX or Kerberos authentication and either UNIX or LDAP authorization mechanisms. Each HPSS server has configuration information that determines the type and level of services available to that server. HPSS software uses these services to determine the caller identity and credentials. Server security configuration is discussed in more detail in Section 5.2: *Server Configuration* of the *HPSS Management Guide*.

Once the identity and credential information of a client has been obtained, HPSS servers enforce access to their interfaces based on permissions granted by an access control list stored in the DB2 table AUTHZACL.

HPSS client interface authentication and authorization security features for end users depend on the interface, and are discussed in the following subsections.

3.9.4.1. Client API

The Client API interface uses either UNIX username/password or Kerberos authentication and either UNIX or LDAP authorization features. Applications that make direct Client API calls must have valid credentials prior to making those calls. Kerberos credentials can be obtained either at the command line level via the **kinit** mechanism or within the application via the **sec_login_set_context** interface. UNIX credentials are determined by the HPSS rpc library based on the UNIX user id and group id of the application process.

3.9.4.2. FTP/PFTP

By default, FTP and Parallel FTP (PFTP) interfaces use either a username/password mechanism or Kerberos credentials to authenticate. Either UNIX or LDAP is used to authorize end users. The end user identity credentials are obtained from the principal and account records in the appropriate security registry.

3.9.4.3. XFS

Since XFS is a filesystem interface, it uses the standard filesystem security mechanisms - owners, groups and UNIX mode bits to enforce security policy. For communication between the HDM and the DMG, the regular HPSS server authentication and authorization mechanisms are used.

3.9.4.4. Name Space

Enforcement of access to HPSS name space objects is the responsibility of the Core Server. A user's access rights to a specific name space object are determined from the information contained in the object's ACL, and the user's credentials.

3.9.4.5. Security Audit

HPSS provides the ability to record information about authentication, file creation, deletion, access, and authorization events. The security audit policy in each HPSS server determines what audit records a server will generate. In general, all servers can create authentication events, but only the Core Server will generate file events. The security audit records are sent to the log file and are recorded as security type log messages.

3.9.5. Logging Policy

The logging policy provides the capability to control which message types are written to the HPSS log files. In addition, the logging policy is used to control whether alarms, events, and status messages are sent to the Storage System Manager to be displayed. Logging policy is set on a per server basis. Refer to Section 9.2.1: *Creating a Log Policy* of the *HPSS Management Guide* for a description of the supported message types.

If a logging policy is not explicitly defined for a server, the default log policy will be applied. The default log policy is selected from the Global Configuration window. If no Default Log Policy entry has been defined, only Alarm and Event messages will be logged. All Alarm, Event, and Status messages generated by the server will also be sent to the Storage System Manager.

The administrator might consider changing a server's logging policy under one of the following circumstances:

- A particular server is generating excessive messages. Under this circumstance, the administrator could use the logging policy to limit the message types being logged and/or sent to the Storage System Manager. This will improve performance and potentially eliminate clutter from the HPSS Alarms and Events window. Message types to disable first would be Trace messages followed by Debug and Request messages.
- One or more servers are experiencing problems which require additional information to troubleshoot. If Alarm, Debug, or Request message types were previously disabled, enabling these message types will provide additional information to help diagnose the problem. HPSS support personnel might also request that Trace messages be enabled for logging.

3.9.6. Location Policy

In past versions of HPSS, the location policy was used to provide the ability to control how often Location Servers in an HPSS installation contacted other servers. The location policy was used to determine how often remote Location Servers were contacted to exchange server location information.

This location policy information is still read by the Location Server, but, in the 6.2 version of HPSS it has no practical value. It will probably be removed in future versions of HPSS.

3.9.7. Gatekeeping

The Gatekeeping Service provides a mechanism for HPSS to communicate information through a well-defined interface to a installation specific customized software policy module. The policy module is placed in a shared library, `/opt/hpss/lib/libgksite.[a|so]`, which is linked into the Gatekeeper. The default policy module does no gatekeeping. If Gatekeeping services are desired in an HPSS installation, this default policy module must be replaced with one that implements the desired policy.

The locally implemented policy module determines which types of requests will be monitored (authorized caller, create, open, and stage). Upon initialization, each Core Server looks for a

Gatekeeper configured in its storage subsystem. If one is found, the Core Server asks the Gatekeeper for its monitor types by calling the **gk_GetMonitorTypes** function which calls the locally implemented **gk_site_GetMonitorTypes** function which determines which types of requests to monitor. This query by the Core Server occurs each time the Core Server connects to the Gatekeeper, which occurs whenever the Core Server or Gatekeeper is restarted. Therefore, if a site wants to change the types of requests to be monitored, the Core Server and Gatekeeper must be restarted.

For each type of request being monitored, the Core Server calls the appropriate Gatekeeping Service API (**gk_Create**, **gk_Open**, **gk_Stage**) passing along information pertaining to the request. This information includes:

Table 5. Gatekeeping Call Parameters

Name	Description	create	open	stage
AuthorizedCaller	Whether or not the request is from an authorized caller. These requests cannot be delayed or denied by the site policy.	Y	Y	Y
BitFileID	The unique identifier for the file.	N/A	Y	Y
ClientConnectId	The end client's connection uuid.	Y	Y	Y
RealmId	The HPSS realm identifier for the user.	Y	Y	Y
GroupId	The user's group identifier	Y	Y	Y
HostAddr	Socket information for originating host.	Y	Y	Y
OpenInfo	Open file status flag (Oflag).	N/A	Y	N/A
StageInfo	Information specific to stage (flags, length, offset, and storage level).	N/A	N/A	Y
UserId	The user's identifier.	Y	Y	Y

Each Gatekeeping Service API will then call the appropriate Site Interface passing along the information pertaining to the request. If the request had AuthorizedCaller set to TRUE, then the Site "Stat" Interface will be called (**gk_site_CreateStats**, **gk_site_OpenStats**, **gk_site_StageStats**) and the Site Interface will not be permitted to return any errors on these requests. Otherwise, if **AuthorizedCaller** is set to **FALSE**, then the normal Site Interface will be called (**gk_site_Create**, **gk_site_Open**, **gk_site_Stage**) and the Site Interface will be allowed to return no error or return an error to either retry the request later or deny the request. When the request is being completed or aborted the appropriate Site Interface will be called (**gk_site_Close**, **gk_site_CreateComplete**, **gk_site_StageComplete**). Examples of when a request gets aborted are when the Core Server goes

DOWN or when the user application is aborted.

NOTES:

1. All open requests to the Core Server will call the Gatekeeping Service open API (**gk_Open**). This includes opens that end up invoking a stage.
2. Any stage call that is invoked on behalf of open will NOT call the Gatekeeping Service stage API (**gk_Stage**). (e.g. The ftp **site stage <filename>** command will use the Gatekeeping Service open API, **gk_Open**, rather than the Gatekeeping Service stage API, **gk_Stage**.)
3. Direct calls to stage (**hpss_Stage**, **hpss_StageCallBack**) will call the Gatekeeping Service stage API (**gk_Stage**).
4. If the site is monitoring Authorized Caller requests then the site policy interface won't be allowed to deny or delay these requests, however it will still be allowed to monitor these requests. For example, if a site is monitoring Authorized Caller and Open requests, then the site **gk_site_Open** interface will be called for open requests from users and the **gk_site_OpenStats** interface will be called for open requests due an authorized caller request (e.g. migration by the MPS). The site policy can NOT return an error for the open due to migration, however it can keep track of the count of opens by authorized callers to possibly be used in determining policy for open requests by regular users. Authorized Caller requests are determined by the Core Server and are requests for special services for MPS and XFS. These services rely on timely responses, thus gatekeeping is not allowed to deny or delay these special types of requests.
5. The Client API uses the environment variable **HPSS_GKTOTAL_DELAY** to place a maximum limit on the number of seconds a call will delay because of **HPSS_ERETRY** status codes returned from the Gatekeeper. See Section 13.1: *Client API Configuration* of the *HPSS Management Guide* for more information.

Refer to *HPSS Programmer's Reference, Volume 1* for further specifications and guidelines on implementing the Site Interfaces.

3.10. Storage Characteristics Considerations

This section defines key concepts of HPSS storage and the impact the concepts have on HPSS configuration and operation. These concepts, in addition to the policies described above, have a significant impact on the usability of HPSS.

Before an HPSS system can be used, the administrator must create a description of how the system is to be viewed by the HPSS software. This process consists of learning as much about the intended and desired usage of the system as possible from the HPSS users and then using this information to determine HPSS hardware requirements and the configuration of the hardware to provide the desired performance. The process of organizing the available hardware into a desired configuration results in the creation of a number of HPSS metadata objects. The primary objects created are classes of service, storage hierarchies, and storage classes.

A Storage Class is used by HPSS to define the basic characteristics of storage media. These characteristics include the media type (the make and model), the media block size (the length of each basic block of data on the media), the transfer rate, and the size of media volumes. These are the physical characteristics of the media. Individual media volumes described in a Storage Class are called Physical Volumes (PVs) in HPSS.

Storage Classes also define the way in which Physical Volumes are grouped to form Virtual Volumes

(VVs). Each VV contains one or more PVs. The VV characteristics described by a Storage Class include the VV Block Size and VV Stripe Width. If PVs are grouped one at a time, so that their Stripe Width is one, they are still defined as VVs.

A number of additional parameters are defined in Storage Classes. These include migration and purge policies, minimum and maximum storage segment sizes, and warning thresholds.

An HPSS storage hierarchy consists of multiple levels of storage where each level is described by a Storage Class. Files are moved up and down the storage hierarchy via stage and migrate operations, based upon storage policy, usage patterns, storage availability, and user requests. If a file is recorded at multiple levels in the hierarchy, the more recent data will be found at the higher level (lowest level number) in the hierarchy.

Class of Service (COS) is an abstraction of storage system characteristics that allows HPSS users to select a particular type of service based on performance, space, and functionality requirements. Each COS describes a desired service in terms of characteristics such as minimum and maximum file size, transfer rate, access frequency, latency, and valid read or write operations. A file resides in a particular COS which is selected when the file is created. Underlying a COS is a storage hierarchy that describes how data for files in that class are to be stored in the HPSS system. A COS can be associated with a fileset such that all files created in the fileset will use the same COS.

The relationship between storage class, storage hierarchy, and COS is shown in Figure 10: *Relationship of Class of Service, Storage Hierarchy, and Storage Class.*

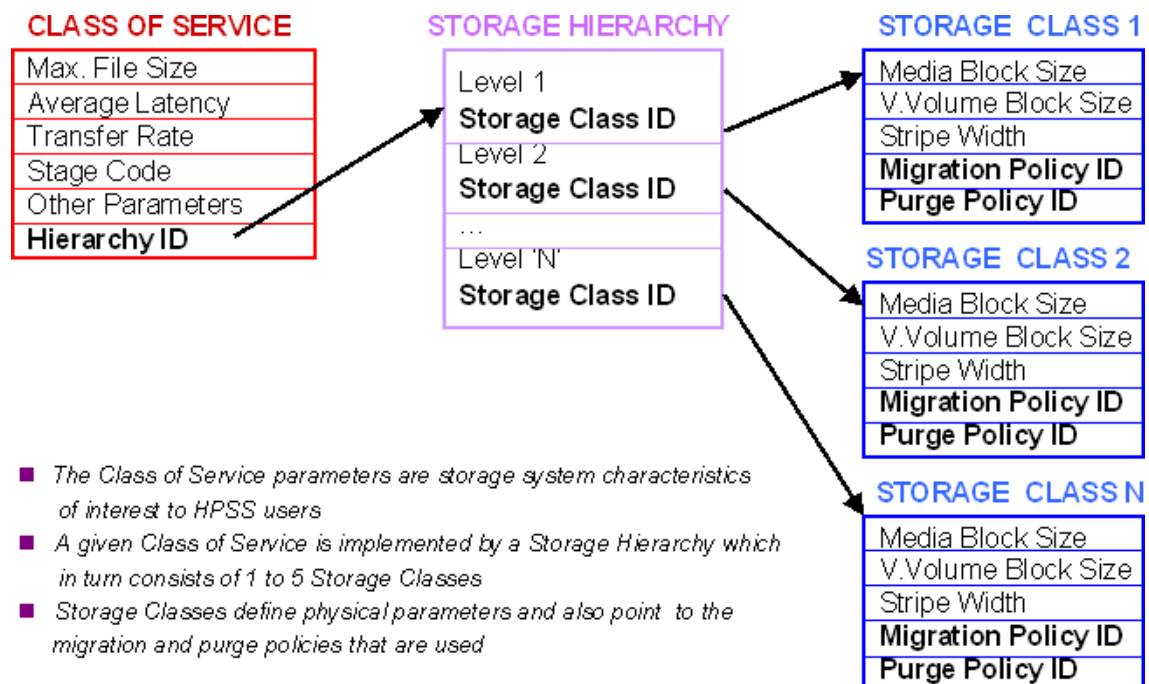


Figure 7. Relationship of Class of Service, Storage Hierarchy, and Storage Class

3.10.1. Storage Class

Each virtual volume and its associated physical volumes belong to some storage class in HPSS. The

SSM provides the capability to define storage classes and to add and delete virtual volumes to and from the defined storage classes. A storage class is identified by a storage class ID and its associated attributes. For detailed descriptions of each attribute associated with a storage class, see Section 6.1.1: *Configuring a Storage Class* of the *HPSS Management Guide*.

The sections that follow give guidelines and explanations for creating and managing storage classes.

3.10.1.1. Media Block Size Selection

Guideline: Select a block size that is smaller than or equal to the maximum physical block size that a device driver can handle.

Explanation: For example, see Section 3.10.1.12: *Some Recommended Parameter Values for Supported Storage Media* on page 106 for recommended values for tape media supported by HPSS.

3.10.1.2. Virtual Volume Block Size Selection (disk)

Guideline: The virtual volume block size must be a multiple of the underlying media block size.

Explanation: This is needed for correct operation of striped I/O. It may be necessary to experiment with combinations of disk and tape VV Block Sizes to find combinations that provide maximum transfer performance

3.10.1.3. Virtual Volume Block Size Selection (tape)

Guideline 1: The VV block size must be a multiple of the media block size

Explanation: This is needed for correct operation of striped I/O.

Guideline 2: Pick an I/O transfer buffer size such that the size of the buffer being used to write this Storage Class is an integer multiple of the block size.

Explanation: Assume files are being written via standard FTP directly into a tape Storage Class. Also assume FTP is set up to use a 4 MB buffer size to write the data. This means that writes are done to the tape with a single 4 MB chunk being written on each write operation. If the tape virtual volume block size is not picked as indicated by the guideline, two undesirable things will happen. A short block will be written on tape for each one of these writes, which will waste data storage space, and the Core Server will create a separate storage segment for the data associated with each write, which wastes metadata space. Performing these extra steps will degrade transfer performance. See also Section 3.10.1.12: *Some Recommended Parameter Values for Supported Storage Media* on page 106 for further information about selecting block sizes.

Guideline 3: Disk and tape VV block sizes should be equal if possible.

Explanation: The system is designed to maximize throughput of data when it is migrated from disk to tape or tape to disk. For best results, the sizes of the VV blocks on disk and tape in a migration path should be the same. If they are different, the data will still be migrated, but the Movers will be forced to reorganize the data into different size VV blocks which can significantly impact performance.

3.10.1.4. Stripe Width Selection

Stripe width determines how many physical volumes will be accessed in parallel when doing read/writes to a Storage Class.

Guideline 1: For tapes, the stripe width should be less than half the number of available tape drives.

Explanation: There must be enough tape drives to support the Stripe Width. The **repack** and **recover**

utility programs copy data from one tape VV to another, so the number of available tape drives of the appropriate type must be at least twice the tape Stripe Width, for these programs to function. Migration of files between tape storage classes in a hierarchy, that are of the same media type, requires at least twice as many available tape drives as the Stripe Width of the Storage Class.

Guideline 2: Select a stripe width that results in data transmission rates to/from the drives matching or exceeding the network transmission rate.

Explanation: Configuring stripe widths that result in transmission rates that exceed the network transmission rate will waste device resources, since more hardware and memory (for Mover data buffers) will be allocated to the transfer, without achieving any performance improvement over a smaller stripe width. Also, if a large number of concurrent transfers are expected, it may be better, from an overall system throughput point of view, to use stripe widths that provide something less than the throughput supported by the network. The aggregate throughput of multiple concurrent requests will saturate the network. Overall throughput will be improved by consuming fewer device and memory resources.

Guideline 3: For smaller files, use a small stripe width or stripe width 1.

Explanation: If writing directly to tape, rather than via a disk cache, writing a file will result in the mounting and positioning of all of the tapes before data transmission can begin. This latency will be driven by how many mounts can be done in parallel, plus the mount time for each physical volume. If the file being transmitted is small, all of this latency could cause performance to be worse than if no striping were used at all.

As an example of how to determine stripe width based on file size and drive performance, imagine a tape drive that can transmit data at about 10 MB/second and it takes about 20 seconds on average to mount and position a tape. For a one-way stripe, the time to transmit a file would be:

$$(\text{<File Size in MB>} / 10) + 20$$

Now consider a 2-way stripe for this Storage Class which has only one robot. Also assume that this robot has no capability to do parallel mounts. In this case, the transmission time would be:

$$(\text{<File Size in MB>} / 20) + (2 * 20)$$

The calculation indicates that the single stripe would generally perform better for files that are less than 400 MB in size.

Guideline 4: Migration can use larger stripe widths.

Explanation: The tape virtual volume is usually mounted and positioned only once when migrating from disk to tape. In this case, larger stripe widths can perform much better than smaller stripe widths.

Guideline 5: The number of drives available for media in this Storage Class should be a multiple of the stripe width.

Explanation: Unless the drives are shared across storage classes, if the number of drives available is not a multiple of the stripe width then less-than-optimal use of the drives is likely.

3.10.1.5. Blocks Between Tape Marks Selection (tape only)

The number of tape physical blocks written between tape marks can be controlled. Tape marks are generated for two reasons: (1) to force tape controller buffers to flush so that the Mover can better determine what was actually written to tape, and (2) to quicken positioning for partial file accesses. In general, larger values for Blocks Between Tape Marks are favored as modern tape drives rely on data

streaming to maximize performance. For recommended values for various media types, see Section 3.10.1.12: *Some Recommended Parameter Values for Supported Storage Media* on page 106.

3.10.1.6. Minimum Storage Segment Size Selection (disk only)

The Core Server maps disk files onto a series of disk storage segments. The size of the storage segments is controlled by the **Min Storage Segment Size** parameter, the **Max Storage Segment Size** parameter, and the **Average Number of Segments** parameter. The smallest amount of disk storage that can be allocated to a file is determined by the **Min Storage Segment Size** parameter. This parameter should be chosen with disk space utilization in mind. For example, if writing a 4 KB file into a Storage Class where the storage segment size is 1,024 KB, then 1,020 KB of the space will be wasted. At the other extreme, each file can use at most 10,000 disk storage segments, so it isn't possible to write a terabyte file to a disk Storage Class with a **Maximum Storage Segment Size** below 128 megabytes. When file size information is available the Core Server will attempt to choose an optimal storage segment size between **Min Storage Segment Size** and **Max Storage Segment Size** with the goal of creating **Average Number of Segments** for the bitfile.

Guideline: Care should be taken when selecting the minimum storage segment size. If data will be migrated from disks in the Storage Class to a tape Storage Class, the value of the Minimum Storage Segment Size parameter should meet one of the following conditions. These rules help prevent the creation of excessive numbers of tape storage segments when files are migrated from disk to tape.

- If the storage segment size is larger than the tape stripe length, it should be an integer multiple of the tape stripe length. The storage segment size may be equal to the tape stripe length.
- If the storage segment size is smaller than the tape stripe length, the tape stripe length should be an integer multiple of the storage segment size, and, it should be not more than 32 times the storage segment size.

Guideline: When a large range of file sizes are to be stored on disk, define multiple disk storage classes with appropriate storage segment sizes for the sizes of the files that are expected to be stored in each Storage Class.

Explanation: The Class of Service (COS) mechanism can be used to place files in the appropriate place. Note that although the Core Server provides the ability to use COS selection, current HPSS interfaces only take advantage of this in two cases. First, the pput command in PFTP automatically takes advantage of this by selecting a COS based on the size of the file. If the FTP implementation on the client side supports the alloc command, a COS can also be selected based on file size. Files can also be directed to a particular COS with FTP and PFTP commands by using the site setcos command to select a COS before the files are stored. When setting up Classes of Service for disk hierarchies, take into account both the **Minimum Storage Segment Size** parameter and the **Maximum Storage Segment Size** parameter in determining what range of file sizes a particular COS will be configured for.

3.10.1.7. Maximum Storage Segment Size Selection (disk only)

This parameter, along with **Minimum Storage Segment Size** and **Average Number of Storage Segments**, is used by the Core Server to optimally choose a storage segment size for bitfiles on disk. The largest storage segment size that can be selected for a file in a Storage Class is limited by this parameter.

Guideline: In order to avoid creating excessive fragmentation of the space on disks in this Storage Class, it is recommended that this be set no higher than 5% of the size of the smallest disk Virtual Volume allocated in this Storage Class.

3.10.1.8. Maximum VVs to Write (tape only)

This parameter restricts the number of tape VVs, per Storage Class, that can be concurrently written by the Core Server. Its purpose is to minimize the number of tape mounts and to limit the number of tape VVs being written, to prevent files from being scattered over a number of tapes. The number of tape drives used to write files in the Storage Class will be limited to approximately the value of this field times the stripe width defined for the Storage Class. Note that this field only affects tape write operations. Read operations are not limited by the value defined by this parameter.

3.10.1.9. Average Number of Storage Segments (disk only)

This parameter, along with Minimum Storage Segment Size and Maximum Storage Segment Size, is used by the Core Server to optimally choose a storage segment size for bitfiles on disk. The Core Server attempts to choose a storage segment size between Storage Segment Size and Maximum Storage Segment Size that would result in creating the number of segments indicated by this field.

Guideline: For best results, it is recommended that small values (< 10) be used. This results in minimizing metadata and optimizing migration performance. The default of 4 will be appropriate in most situations.

3.10.1.10. PV Estimated Size / PV Size Selection

Guideline: For tape, select a value that represents how much space can be expected to be written to a physical volume in this Storage Class with hardware data compression factored in.

Explanation: The Core Server uses this value as a guide in selecting tapes for writing, but regards it as an estimate only. Regardless of its value, the tape will be filled before another tape is chosen for writing.

Rule 1: For disk, the PV Size value must be the exact number of bytes available to be written on the PV. This value must be a multiple of the media block size and the VV block size. It may be necessary to round the actual size of the volume down to one of these multiples. The SSM will enforce these rules when the window fields are filled in.

Rule 2: For disk, the PV Size value must be less than or equal to the Bytes on Device value described in Section 7.1: *Configure a New Device and Drive* of the *HPSS Management Guide*.

3.10.1.11. Optimum Access Size Selection

Guideline: Generally, a good value for Optimum Access Size is the Stripe Length, which is the Virtual Volume Block Size times the Stripe Width.

Explanation: This field is advisory in nature in the current HPSS release. In the future, it may be used to determine buffer sizes. Generally, a good value for this field is the Stripe Length; however, in certain cases, it may be better to use a buffer that is an integer multiple of the Stripe Length. The simplest thing at the present time is to set this field to the Stripe Length. It can be changed in the future without complication.

3.10.1.12. Some Recommended Parameter Values for Supported Storage Media

Table 5 and Table 6 contain suggested values for storage resource attributes based on the media type. The given values are not the *only* acceptable values, but represent reasonable settings for the various media types. See Section 3.9.6: *Location Policy* on page 99 for more information about setting the storage characteristics.

3.10.1.12.1. Disk Media Parameters

Table 7 contains attributes settings for the supported disk storage media types.

Table 6. Suggested Block Sizes for Disk

Disk Type	Media Block Size	Minimum Access Size	Minimum Virtual Volume Block Size
SCSI Attached	4 KB	0	1 MB
SSA Attached	4 KB	0	1 MB
Fibre Channel Attached	4 KB	0	1 MB

In Table 7:

- Media Block Size is the block size to use in the Storage Class definition. For disk, this value should also be used when configuring the Mover devices that correspond to this media type. Note that this value will not limit the amount of data that can be read from or written to a disk in one operation—it is used primarily to perform block boundary checking to ensure that all device input/output requests are block aligned. This value should correspond to the physical block size of the disk device.
- Minimum Access Size is the size of the smallest access request that should regularly be satisfied by the media type. The performance of smaller accesses will be seriously degraded. A value of zero indicates that the media is suitable for supporting all data accesses.
- Minimum Virtual Volume Block Size is the smallest block size that should be configured for virtual volumes of the media type. Smaller values will cause increased overhead when transferring to or from volumes configured with Stripe Widths greater than one. Virtual Volume Block Size has little or no effect on virtual volumes whose Stripe Width is one.

Note: When SCSI, SSA or Fibre Channel attached disks are combined to form striped virtual volumes, the minimum access size should become, at a minimum, the stripe width of the virtual volume multiplied by the virtual volume block size. If not, data access will only use a subset of the striped disks and therefore not take full advantage of the performance potential of the virtual volume.

3.10.1.12.2. Tape Media Parameters

The following table contains attributes settings for the supported tape storage media types.

Table 7. Suggested Block Sizes for Tape

Tape Type	Media Block Size	Blocks Between Tape Marks	Estimated Physical Volume Size
Ampex DST-312	1 MB	1024	50, 150, 330 GB
Ampex DST-314	1 MB	1024	100, 300, 660 GB
IBM 3580 (LTO)	256 KB	1024	100 GB
IBM 3580 (LTO Gen 2)	256 KB	1024	200 GB
IBM 3580 (LTO Gen 3)	256 KB	2048	400 GB
IBM 3580 (LTO Gen 4)	256 KB	2048	800 GB
IBM 3590	256 KB	512	10, 20 GB
IBM 3590E	256 KB	512	20, 40 GB
IBM 3590H	256 KB	512	60, 120 GB
IBM 3592 J	256 KB	1024	60 GB (JJ Short), 300 GB (JA Standard)
IBM 3592 E05	256 KB	2560	100 GB (E05 Short), 500 GB (E05 Standard), 660 GB (E05 XL)
IBM 3592 E06	256 KB	2560	100 GB (E06 Short), 500 GB (E06 Standard), 1000 GB (E06 XL)
Sony GY-8240	256 KB	1024	60, 200 GB
Sony SAIT-1	256 KB	1024	500 GB
StorageTek 9840A	256 KB	1024	20 GB
StorageTek 9840B	256 KB	1024	20 GB
StorageTek 9840C	256 KB	1024	40 GB
StorageTek 9840D	256 KB	1024	75 GB
StorageTek 9940A	256 KB	1024	60 GB
StorageTek 9940B	256 KB	1024	200 GB

StorageTek T10000A	256 KB	2560	500 GB
StorageTek T10000B	256 KB	2560	1000 GB

In the above table:

- Media Block Size is the block size to use in the Storage Class definition. This is the size of the data blocks written to tape. Note that for tape devices, the Mover configuration does not contain the Media Block Size. This value may have a significant impact on data transfer performance, as for most tape devices each input/output request must be for the media block size. If a large block size is used for relatively small write requests, space may be wasted on the tape.
- Blocks Between Tape Marks is the number of media blocks to be written between tape marks. A relatively small value has the benefit of shorter positioning times to the middle of files. Small values have the penalties of poorer media utilization and lower performance when writing tapes. Since files are usually read in their entirety, and modern tape controllers employ sophisticated positioning logic and are designed to stream data to tape, larger values of the Block Between Tape Mark parameter are recommended.
- Estimated Physical Volume Size is the estimated size of the physical volumes to be set in the Storage Class definition. These values are based on the expected media to be used with the specified type. In some cases, different length tape media may be used, which may have an effect on the estimated size for a given physical volume (e.g., regular or extended length 3480/3490 format cartridges). Note that the values listed do not take into account any data compression that may be performed by the tape drive. Also note that this value is for informational purposes only and does not affect the amount of user data written to a tape volume by the Core Server. The server fills each tape Virtual Volume such that the amount of data written to the tape varies with the compressibility of the data.

3.10.2. Storage Hierarchy

Each HPSS file is stored in a storage hierarchy consisting of an ordered list of storage classes. A storage hierarchy can have up to 5 levels starting with level 0. The highest level (first level written to) is always level 0 and the lowest is level 4. Storage Classes are expected to be arranged in a hierarchy in order of descending performance. For example, a level 0 Storage Class could be fast disk while a level 4 Storage Class could be a slower, large capacity tape system. The SSM provides a means to define storage hierarchies. A storage hierarchy is identified by a storage hierarchy ID and its associated attributes. For detailed descriptions of each attribute associated with a storage hierarchy, see Section 6.2.1: *Configuring a Storage Hierarchy* of the *HPSS Management Guide*. The following is a list of rules and guidelines for creating and managing storage hierarchies.

Rule 1: All writes initiated by clients are directed to the highest level (level 0) in the hierarchy.

Rule 2: The data of a file at a level in a hierarchy is associated with a single Core Server.

Rule 3: Parts or all of a file may appear at multiple levels in a storage hierarchy. If data for a file does appear at multiple levels of the hierarchy, the data at the higher level is always the more recent data.

Rule 4: Migration of data does not skip levels in the hierarchy, except in the special case of creating duplicate copies when doing disk migration.

Rule 5: The client stage command can only stage data to the top level (level 0) in the hierarchy.

Rule 6: A given Storage Class can only occur once in the same hierarchy.

3.10.3. Class of Service

Each HPSS file belongs to a single Class of Service (COS) which is selected when the file is created. It is selected via Class of Service Hints information passed to the Core Server when the bitfile is created. If using the Client API, the application program has full access to this hints information. FTP users can use the “quote” command to set the COS. A **pput** request in PFTP automatically selects a COS based on file size unless the user explicitly selects the COS.

The SSM provides a means to define Classes of Service. A COS is identified by a COS ID and its associated attributes. For detailed descriptions of each attribute associated with a Class of Service, see Section 6.3: *Classes of Service* of the *HPSS Management Guide*.

The Force Selection flag can be set in the COS definition to prevent automatic selection. If this flag is set, the COS can only be selected by asking for the COS by ID or Name.

The sections that follow give guidelines and explanations for creating and managing classes of service.

3.10.3.1. Selecting Minimum File Size

Guideline: This field is used to indicate the smallest file that should be stored in the COS.

Explanation: This limit is not enforced and is advisory in nature. The minimum file size can be used as a criteria for selecting a COS via the COS hints mechanism. Currently, PFTP and FTP clients that support the **alloc** command will use the size hints when creating a new file. Ensure that the Minimum File Size is less than the Maximum File Size.

3.10.3.2. Selecting Maximum File Size

Guideline: This field can be used to indicate the largest file that may be stored in the COS.

Explanation: If the **Enforce Max File Size** option is selected, an attempt to perform an operation on a file that would cause this value to be exceeded will be rejected. The underlying storage hierarchy should be set up so that the defined storage classes support files of this size in a reasonable fashion. For details, see Sections 3.10.1: *Storage Class* and 3.10.2: *Storage Hierarchy* on storage classes and storage hierarchies. This field can be used via the COS Hints mechanism to affect COS selection. PFTP and FTP clients that support the **alloc** command will use the size hints when creating a new file. Ensure that the Minimum File Size and the Maximum File Size do not overlap; otherwise the data placement may be indeterminate.

3.10.3.3. Selecting Stage Code

This field determines whether a file is to be staged to the highest level in the storage hierarchy when the file is opened by one of the Client API function calls to the Core Server. This field can be used via the COS Hints mechanism to affect COS selection. The valid options are as follows:

Guideline 1: Select the **No Stage** option if staging of files is not desired.

Explanation: Data read from the file may come from lower levels in the storage hierarchy if the data does not exist at the top level. This option is normally selected if the top level in the hierarchy is not disk or if the users of files stored in this COS wish to control staging directly via user stage requests.

Guideline 2: Select the **Stage on Open** option to stage the entire file to the top level of the hierarchy synchronously. The Client API open operation will block while the file is being staged.

Explanation: This option is commonly selected when the top level of the hierarchy is disk and the files in this Class of Service are small to moderate in size. Use this option if you want to be

guaranteed that the file is completely and successfully staged before it is read. If the stage operation fails, the open will return with an error.

Guideline 3: Select the **Stage on Open Async** option if you wish to stage the entire file to the top level in the hierarchy and do not want the Client API open to block.

Explanation: When this option is selected, the file is staged in sections and the read and write operations that access this file are blocked only until the portion of the file they reference has been completely staged. Normally, this option is selected when the top level of the hierarchy is disk and the files in this COS are fairly large in size. This option is only available when the top level in the hierarchy is disk. If the top level is tape and this option is specified, the **No Stage** option will be used instead.

Guideline 4: Select the **Stage on Open Background** option if you want the stage operation to be queued in the Core Server and processed by a background Core Server thread.

Explanation: The Client API open request will return with success if the file is already staged. Otherwise, a stage request is placed in a queue and will be processed by the Core Server in the background. A busy error is returned to the caller. This option allows a large number of stages (up to 2000) to be queued in the Core Server and processed as resources permit. The other stage options will result with a busy error if Core Server threads are not immediately available to process the request.

Guideline 5: Select the **Retry Stage Failures from Secondary Copy** option if you want to configure a Storage Class so that stage failures of the primary copy can be automatically retried from a valid second copy. The associated hierarchy must first be configured for multiple copies.

Explanation: When a stage from the primary copy fails and a secondary copy of the file is available, HPSS will usually reissue the stage operation from the secondary copy. This is typically done when a tape, holding the primary copy, becomes damaged and cannot be read. A warning that the stage operation has used the secondary copy will appear in the SSM Alarms and Events window.

3.10.3.4. Selecting Optimum Access Size

This field is only advisory in nature; however, for later releases it may be used by interfaces to dynamically select good buffer sizes.

Guideline 1: Generally, if the file is being staged on open, **Optimum Access Size** should be set to the same value that **Optimum Access Size** is set to in the Storage Class at the top of the hierarchy.

Guideline 2: If data is not being staged to the top level before it is read (either automatically or by user command), select a value that is an integer multiple of the largest **Optimum Access Size** field found among the storage classes that make up this hierarchy.

3.10.3.5. Selecting Average Latency

This field can be used via the COS Hints mechanism to affect COS selection.

Guideline 1: This field should generally be set to the value of the **Average Latency** field in the Storage Class at the top level of the hierarchy when the **Stage on Open** option is in effect. If files are usually accessed only once after they are staged, the average latency should be set to the latency of the level they are staged from.

Guideline 2: If it is expected that most of the requests for files in this COS are read requests, then it may be best to set the value of this field equal to the **Average Latency** field in the Storage Class in the hierarchy where most of the data accesses come from.

Guideline 3: If files are written into the COS much more frequently than read, use the **Average**

Latency field from the Storage Class at the top level in the hierarchy.

3.10.3.6. Selecting Transfer Rate

This field can be used via the COS Hints mechanism to affect COS selection.

Guideline 1: This field should generally be set to the value of the **Transfer Rate** field in the Storage Class that is at the top level in the hierarchy. This should always be the case if the data is being staged on open.

Guideline 2: If a large percentage of the reads are being done from a lower level in the hierarchy, consider setting the transfer rate based on the **Transfer Rate** associated with the Storage Class at this lower level.

3.10.3.7. StripeLength and StripeWidth Hints

These fields can be used via the COS Hints mechanism to affect COS selection.

Guideline: StripeLength and StripeWidth hints are available in the hints mechanism. When specified in the hints, StripeLength and StripeWidth from the Storage Class at the top level of each hierarchy are used in the COS selection algorithm.

3.10.4. File Families

A file family is an attribute of an HPSS file that is used to group a set of files on a common set of tape virtual volumes. HPSS supports file families only on tape volumes. In addition, families can only be specified by associating a family with a fileset, and creating the file in the fileset. When a file is migrated from disk to tape, it is migrated to a tape virtual volume assigned to the family associated with the file. If no tape VV is available for writing in the family, a blank tape is reassigned from family zero to the file's family. The family affiliation is preserved when tapes are repacked. Each file in HPSS is assigned a family designation. The default family is family zero, which is interpreted by HPSS as no family affiliation.

HPSS places no restriction on the values assigned to the File Family IDs, other than the special meaning of family zero. A name must be assigned to each file family.



Defining multiple file families may have an impact on system migration performance. MPS may have to mount a significantly larger number of tapes to complete a migration from a disk Storage Class if the files are spread across a large number of file families, compared to the number of mounts that would be required if all the files were in the same family.

3.11. HPSS Performance Considerations

This section provides some additional hints for enhancing HPSS performance.

3.11.1. DB2

Because all HPSS operations involve DB2, it is important to optimized DB2 performance. There are a number of configuration settings that can greatly affect DB2 performance. Some of these are discussed in Section 5.7: *Tune DB2* on page 171. For a detailed discussion on tuning DB2, refer to the *DB2 Performance Tuning Guide*, an IBM redbook.

The following is a list of areas to consider when optimizing DB2 performance for HPSS:

- Database usage
 - Average number of users and applications connected to DB2

- Maximum users and applications connected to DB2
- Nature of usage: read or update
- Database logging
 - Hardware or software mirroring
 - Disk speed and reliability: select the fastest, most reliable disk
 - Location of physical disks and database data: it is recommended that they be separate
- Database recovery
 - Enabling dropped table recovery will decrease database performance
- Amount of available memory and size of buffer pools
 - Buffer pools should not be larger than the tables using them (i.e not over-allocating memory)
 - Increase buffer pool size (i.e. increase memory available for metadata)
- Type of node
 - Amount of cache, memory, CPU available
- HPSS/DB2 server proximity
- Storage media

3.11.2. Bypassing Potential Bottlenecks

HPSS performance is influenced by many factors, such as device and network speeds, configuration and power of HPSS server nodes, DB2 configuration, storage resource configuration, and client data access behavior.

HPSS provides mechanisms to bypass potential data transfer performance bottlenecks, provided that the system configuration provides the additional resources necessary. For example, if the performance of a single disk device is the limiting factor in a transfer between HPSS and a client application, the disks can be reconfigured in a striped Storage Class to allow parallel transfers at higher transfer rates. If after forming the stripe group, the I/O or processor bandwidth of a single node becomes the limiting factor, the devices can be distributed among a number of nodes, alleviating the limitations of a single node.

If the client node or single network between the client and HPSS becomes the limiting factor, HPSS supports transferring data to or from multiple client nodes in parallel, potentially using multiple physical networks, to bypass those potential bottlenecks.

During system planning, consideration should be given to the number and data rates of the devices, node I/O bandwidth, network bandwidth, and client node bandwidth to attempt to determine a configuration that will maximize HPSS performance given an anticipated client work load.

3.11.3. Configuration

The configuration of the HPSS storage resources (see Section 3.9.6: *Location Policy*) is also an important factor in overall HPSS performance, as well as how well the configuration of those resources matches the client data access patterns.

For example, if a site provides access to standard FTP clients and allows those clients to write data

directly to tape, the buffer size used by the FTP server and the virtual volume block size defined for the Storage Class being written to will have a significant impact. If the buffer size used by the FTP server is not a multiple of the virtual volume block size, each buffer written will result in a distinct storage segment on the tape. This will cause additional metadata to be stored in the system and extra synchronization processing of the tape. However, if the buffer size is a multiple of the virtual volume block size, each write will continue to append to the same storage segment as the previous write. This will continue until the final write for the file, which will usually end the segment, thus reducing the amount of metadata generated and media processing.

3.11.4. FTP/PFTP

Data transfers performed using the standard FTP interface are primarily affected by the buffer size used by the FTP Daemon. The buffer size can be configured as described in Section 13.2: *FTP Daemon Configuration* of the *HPSS Management Guide*. It should be a multiple of the storage segment size, if possible. Otherwise, it should be at least a multiple of the virtual volume block size. If the buffer size is too small, the FTP Daemon will need to issue a large number of individual read or write requests; however, if the buffer size is too large, the FTP Daemon will require a large amount of memory, which may cause additional paging activity on the system.

The size of the FTP Daemon buffer is extremely important if the FTP clients write files directly to a tape Storage Class, as described in Section 3.11.3: *Configuration* on page 113.

Parallel FTP (PFTP) uses TCP/IP to move data.

Note that the PFTP data transfer commands (e.g., **pput** and **pget**) are not influenced by the FTP Daemon buffer size because the data flows directly between the client and Movers.

Note that PFTP clients that use the standard FTP data transfer commands (e.g., **put** and **get**) have the same performance considerations as standard FTP clients.

Parallel transfers move the data between the Mover and the end-client processes bypassing the HPSS FTPD. Users should be educated to use the parallel functions rather than the non-parallel functions. NOTE: ASCII transfers are not supported by the parallel functions and the non-parallel functions will need to be specified for ASCII transfers. ASCII transfers are NOT typically required, but the end-customer should familiarize themselves with the particulars.

Parallel transfers should be optimized so that the Class of Service (COS), Media Stripe Widths, Network Stripe Widths, and Parallel Block Sizes are consistent with each other. For example, using a Network Stripe Width of 4 with a Media Width of 2 may result in poorer performance than if both specifications are equal. Specifying a Network Stripe Width of 4 where there is only one network interface may not provide any improvement over a lower Network Stripe Width (2) if the bandwidth of the network is (over-)filled by a 4-way stripe.

Non-parallel transfers occur via a “stage and forward” approach (Device <==> Mover <==> HPSS FTP Daemon <==> FTP Client.) It is recommended that the “-h” option be specified on the `hpss_pftpd` and that the “hpss_option HOST hostname” be specified in the `ftpaccess` file if the FTP Daemon system has multiple interfaces. The hostname should refer to the highest speed interface available for transmission of data between the FTP Daemon and HPSS Movers. NOTE: this interface MUST be available to all client systems that contact the FTP Daemon.

Where reasonable, the standard FTP ports should be modified to something other than 20/21 on the system acting as the HPSS FTP Daemon. The HPSS FTP Daemon should be set to use the 20/21 ports by default. This reduces the problem of requiring the end-customer to know which port to use for transfers to HPSS. In conjunction with this, it is highly recommended that the `{ftpbanner}` file be used with an appropriate message to provide information to the end-customer that they are accessing HPSS as opposed to a standard system.

3.11.5. Client API

The Client API provides the capability to perform data transfer of any size (the size being parameters supplied by the client to the read and write interfaces). The size of the data transfers can have a significant impact on the performance of HPSS. In general, larger transfers will generate less overhead than a series of smaller transfers for the same total amount of data.

The size of the transfers is extremely important because the clients may write files directly to a tape Storage Class, as described in Section 3.11.3: *Configuration* on page 113.

3.11.6. Core Server

Minor performance degradations may be seen when the Core Server is processing path names with a large number of components, servicing requests for objects which have a large number of Object ACL entries, and servicing create requests for objects which have Initial Container/Initial Object ACL entries.

3.11.7. Location Server

The Location Policy defined for a site generally determines how the Location Server will perform and how it will impact the rest of the HPSS system. View the help for the fields on this screen to determine if the values need to be changed. The default policy values are adequate for the majority of sites. Usually, the only time the policy values need to be altered is when there is unusual HPSS setup.

The Location Server itself will give warning when a problem is occurring by posting alarms to SSM. Obtain the information for the Location Server alarms listed in the *HPSS Error Manual*. To get a better view of an alarm in its context, view the Location Server's statistics screen.

If the Location Server consistently reports a heavy load condition, increase the number of request threads and recycle the Location Server. Remember to increase the number of threads on the Location Server's basic server configuration screen as well. If this doesn't help, consider replicating the Location Server on a different node. Note that a heavy load on the Location Server should be a very rare event.

3.11.8. Logging

Excessive logging by the HPSS servers can degrade the overall performance of HPSS. If this is the case, it may be desirable to limit the message types that are being logged by particular servers. The Logging Policy can be updated to control which message types are logged. A default Log Policy may be specified to define which messages are logged. Typically, Trace, Security, Accounting, Debug, and Status messages are not logged. Other message types can also be disabled. Once the Logging Policy is updated for one or more HPSS servers, the Log Clients associated with those servers must be reinitialized.

3.11.9. Cross Realm Trust

Cross Realm Trust should be established with the minimal reasonable set of cooperating partners. Excessive numbers of Cross Realm connections may diminish security and cause performance problems due to Wide Area Network delays. The communication paths between cooperating realms should be reliable.

3.11.10. Gatekeeping

Sites may choose to implement site policy in the Gatekeeper for load balancing create, open, and/or

stage requests. The site policy could limit the maximum number of non-Authorized Caller requests allowed at once by either delaying or denying particular requests. To delay the request, the site policy may return a special retry status along with the number of seconds to wait before the Client API retries the request. Delaying requests should limit the number of create, open, and/or stage requests performed at a particular point in time, thus decreasing the load on the system. However, care must be taken to figure out the best retry wait scheme to meet the requirements for each site and to configure the correct number of Gatekeepers if the load on one Gatekeeper is heavy. (Note: The maximum number of Gatekeepers per storage subsystem is one.) Also, sites need to write their Site Interfaces optimally to return in a timely manner.

Two special error status codes (**HPSS_ETHRESHOLD_DENY** and **HPSS_EUSER_DENY**) may be used to refine how a site may deny a create, open, or stage requests. If the Core Server receives either of these errors, then it will return this error directly to the Client API rather than performing a retry. Errors other than these two or the special **HPSS_ERETRY** status will be retried several times by the Core Server. See either volume of the *HPSS Programmer's Reference* for more information.

Create, open, and stage requests from Authorized Callers (MPS and XFS) can NOT be delayed or denied due to timing sensitivity of the special requests these servers make to the Core Server. For example, migration of a file by MPS is an Authorized Caller Open request. The site policy could keep track of Authorized Caller requests to further limit non-AuthorizedCaller requests.

If a Gatekeeper is being used for Gatekeeping Services, then the Core Server for each storage subsystem configured to use a particular Gatekeeper will return errors for the create, open, and/or stage requests being monitored by that Gatekeeper when that Gatekeeper is down. For example, if storage subsystem #2 is configured to use Gatekeeper #2, and Gatekeeper #2 is monitoring open requests and is DOWN, then each open by the Core Server in storage subsystem #2 will eventually fail after retrying several times.

3.11.11. XFS

XFS is one of the better performing filesystems available for Linux – particularly when manipulating large files. The XFS filesystem configuration required for use with HPSS must have the XDASM (DMAPI) kernel extension enabled. This adds some additional overhead. However, timing tests indicate that the amount of additional filesystem processing time introduced by DMAPAPI is minimal.

The HPSS HDM handles the namespace, data and administrative events generated by the XFS filesystem. The HDM was designed to introduce very little additional processing when handling namespace events (file creations, renames or deletions). Data events (reads and writes) require communication with the DMAP gateway which means one or more RPCs will be executed. Additionally, attempting to read or modify a file which has been purged from the XFS Filesystem will cause the process to block until the file has been staged back from HPSS.

The HDM keeps an internal record of migration and purge candidates and is capable of quickly completing migration and purge runs which would otherwise take a good deal of time. This makes a 'migrate early, migrate often' strategy a feasible way to keep XFS disks clear of inactive data. It is also possible to configure a minimum file size for migration candidates. This can be used to keep files below a certain size on the XFS filesystem, improving small file access times.

If the HDM appears to be introducing a performance bottleneck, it's possible to configure multiple HDMs on multiple machines to distribute the load.

3.11.12. HPSS VFS Interface

Please refer to Section 1.4: *HPSS VFS Interface Configuration* of the *HPSS Management Guide*.

3.12. HPSS Metadata Backup Considerations

This section contains guidelines for proper maintenance of the HPSS metadata stored in DB2.



The policies described should be fully understood and implemented to protect the HPSS metadata. Failure to follow these policies can lead to unrecoverable data loss.

The remainder of this section is a set of rules associated with backing up HPSS metadata. Though automated archive software like Tivoli Storage Manager is used to backup and protect DB2 data, it is important that each site review this list of rules and check to ensure that their site's backup is consistent with these policies.

When deciding on the size and number of disks for metadata, keep in mind the following:

1. At a minimum, for subsystem-specific metadata, the disks hosting the DB2 instance installation path (i.e the home directory of the instance owner) and the databases' tablespaces should be separate. This is not critical for configuration metadata since it changes infrequently.
2. Ideally, several physical disks should be available for DB2 tablespaces (namespace data can be on one, bitfile data on another, etc.).
3. If the backup software employed involves some sort of disk staging area, this should be on a separate disk, and a large amount of disk should be allocated for this purpose.

For reliability and availability reasons, the disk hosting the instance installation data should be mirrored. The disks hosting the tablespaces should also be mirrored if possible. For performance and reliability reasons, mirroring should be done using separate physical devices (as opposed to the AIX logical volume being mirrored to the same physical drive).

3.13. HPSS Security Considerations

The security requirements between sites differ widely. The HPSS System Administrators must be aware of the sites security requirements and should be aware of the security configuration required in HPSS. Sites should contact their site representative if they have questions regarding HPSS security. For more information on security, see Chapter 2: *Security and System Access* of the *HPSS Management Guide*.

Chapter 4. System Preparation

This section covers the steps that must be taken to appropriately prepare your system for installation and configuration of HPSS and its infrastructure.

- General setup (Section 4.1)
- Setup filesystems (Section 4.2)
- Setup tape libraries (Section 4.3)
- Verify tape drives (Section 4.4)
- Setup disk drives (Section 4.5)
- Setup network parameters (Section 4.6)



To understand and optimize the operation of HPSS, some baseline measurement, evaluation, and tuning of the underlying network and IO subsystems is necessary. Appropriate tuning of these components is necessary for HPSS to perform as expected. Any one component that is not performing at its optimal rate will have an adverse affect on the performance of the entire system. The steps and tools listed in this chapter will help a site make the best use of their available resources. The measurements taken should be saved for future reference. If performance problems occur later on, these values can be compared with new measurements to determine if the performance problems are related to changes in the subsystems. Though disk and tape configurations rarely change without an administrator's knowledge, networks can “mysteriously” degrade for a variety of reasons. This is especially true for client access to the HPSS system.

4.1. General Setup

- Each HPSS administrator should request a login id and password for the IBM HPSS web site at <http://www.hpss-collaboration.org/hpss/IDRequired.jsp>.
- Download copies of the Installation Guide and Management Guide for the appropriate version of HPSS. It will probably be useful to print copies of these documents and keep them handy while installing HPSS.
- Install, configure, and verify the correct prerequisite operating system version on all HPSS, Kerberos client and/or server, and DB2 nodes.
- Check the format of /etc/hosts. If /etc/hosts is used to augment DNS, fully qualified hostnames must be listed first. e.g.,

```
123.45.6.789    host1.domain.gov    host1
```

- Verify the current operating system level:

```
% uname -a
```
- Create appropriate HPSS UNIX user accounts. The hpss user ID and hpss group ID should be created at this time.
- Install the prerequisite software for Kerberos, C compiler, Java, Perl, SSH, and any other specific software that will be used by HPSS. Verify the correct patch levels are/will be installed. Refer to Section 3.3: *Prerequisite Software Considerations* on page 58 for additional information.

- Configure the Perl prerequisite software on HPSS nodes.
- Configure the SSH prerequisite software on the core HPSS server node (at a minimum) and configure SSH to accept connections from IBM Houston. Include the Houston subnet IP addresses 192.94.47 and 12.39.169 in the local firewall routing rules, as necessary.
- Obtain the HPSS deployment tool package from your customer support representative and install it on each HPSS node in `/opt/hpss/tools/deploy`. Run and become familiar with the `lsnode` tool, which will be helpful in other steps.

To run `lsnode` and save the output to `/var/hpss/stats/lsnode.out`:

```
% mkdir -p /var/hpss/stats
% cd /opt/hpss/tools/deploy/bin
% lsnode > /var/hpss/stats/lsnode.out
```

- Obtain your HPSS Realm Id from IBM. This information will be needed when `mkhpss` is used to configure HPSS. For an existing installation, this is the ID which was previously referred to as the DCE Cell ID.

4.2. Setup Filesystems

The following sections describe how to set up the various filesystems used by HPSS and DB2.

4.2.1. DB2 Filesystem

Each database in DB2 has its own log. We recommend that these logs be placed on a filesystem reserved exclusively for this purpose. This should be a fast, but more importantly, stable disk, preferably a RAID device. For example, the following filesystems might be created for a configuration database and subsystem database:

```
/db2/log/cfg
/db2/log/subsys1
```

The `NEWLOGPATH` database configuration variable can be used to direct the database logs to these filesystems.

Additionally, the following filesystems are required for storing the mirrored copy of the DB2 logs:

```
/db2/mirror-log/cfg
/db2/mirror-log/subsys1
```

The DB2 log mirroring is configured using the `MIRRORLOGPATH` variable.

The home directory of the DB2 instance owner should be created as a separate file system on each HPSS node that runs a database instance. By default, the DB2 instance owner is “`hpssdb`” and its home directory is `/var/hpss/hpssdb`.

If the backup utilities included with HPSS will be used for backing up DB2, a filesystem should be created to serve as the disk storage area where backup data is placed before being archived to tape. For example:

```
/db2/backups/cfg
/db2/backups/subsys1
/db2/backups/ldap
/db2/mirror-backups/cfg
```


`/db2/mirror-backups/subsys1`

4.2.2. HPSS Filesystem

Configure `/var/hpss` as a separate file system on each HPSS server node. This filesystem will store HPSS configuration files, log files, MPS reports, and other HPSS related files. It is recommended that this filesystem be at least 1GB in size.

Configure `/var/hpss/adm/core` as a separate file system on each HPSS server node. If an HPSS process fails and creates a core file, it will be placed in this location. It is recommended that this filesystem be configured with at least 2GBs of disk space on Server nodes and 1 GB on Mover nodes.

Configure `/var/hpss/hpssdb` as a separate file system on the HPSS core server node. This filesystem stores the HPSS DB2 instance configuration information as well as the 'CFG' database tables. This filesystem should be created with at least 1GB of disk storage and, like many of the other filesystems related to HPSS, it should be monitored for fullness periodically by the administrator.

Configure `/opt/hpss` as a separate filesystem on the each of the HPSS server and mover nodes. This directory contains the HPSS executables, documentation, libraries, and, for those sites with source code, the development environment to build the HPSS system. For most sites, this filesystem can be configured with 1GB of space. For those sites with source code, at least 2GBs are required.

4.3. Setup Tape Libraries

4.3.1. Special LTO Considerations

When using the LTO PVR, there are special considerations for tape placement.

Storing HPSS and non-HPSS tapes within the same logical library is not recommended. If a non-HPSS application moves tapes around within the library, it can confuse the HPSS PVR about the state of the LTO tape library map (which keeps track of tape locations and available slots). This will cause the map to be re-initialized each time the PVR queries an inconsistent slot, which can cause a significant performance slowdown.

If the logical library must be shared between HPSS and non-HPSS tapes, the following steps should be taken:

- HPSS tapes should be stored in lower numbered slots than the non-HPSS tapes.
- Leave enough empty slots after the HPSS tapes to hold any future HPSS tapes added to this library. The maximum number of HPSS tapes the library may hold is specified in HPSS in the "Cartridge Capacity" field of the Specific Config portion of the LTO PVR server configuration associated with the library.
- Care should be taken so that the non-HPSS applications avoid using slots that have been used for HPSS tapes.
- The tape drive should be locked within HPSS before it is used by a non-HPSS application

If the map becomes inconsistent there may be an occasional error or warning message displayed by the PVR. The message will be similar to: "(d) Invalid parameters passed to LTO Library". This indicates that the PVR is re-reading the library map. The problem is self correcting and should not require outside intervention, beyond issuing a "Mark Repaired" on the affected servers.

4.3.2. IBM 3584

If using an IBM 3584 tape library, install the Atape driver, configure the SCSI Medium Changer

(SMC) library device on the node that will run the HPSS LTO PVR, and verify that it is operational.

To configure the library:

```
root% cfmgr
root% lsdev -Cc tape | grep smc
smc0 Available 40-58-00-0,1 IBM 3584 Library Medium Changer
```

To test communication with the library:

```
% tapeutil -f <smc device filename> inventory
```

To test tape mounts:

```
% tapeutil -f <smc device filename> move <Source Slot> <Drive Slot>
```

To test tape dismounts:

```
% tapeutil -f <drive device filename> unload
% tapeutil -f <smc device filename> move <Drive Slot> <Source Slot>
```

Perform these tests before starting HPSS as the SMC device may be opened by only one process at a time.

For drives in an IBM 3584 robot, identify the robot-specific device id (LTO drive locations) for each Mover tape device. This will be required when configuring the tape drives within HPSS.

To identify robot-specific device ids for each Mover tape device in an LTO robot:

```
% tapeutil -f <smc device filename> inventory
```

Look for addresses of drive devices in the output.

Refer to Section 5.2.8.4: *LTO PVR Specific Configuration* of the *HPSS Management Guide* for more information.

4.3.3. 3494

For a 3494 tape library:

- Identify the robot-specific device id for each Mover tape device. This will be required when configuring the tape drives within HPSS.

To identify robot-specific device id for each Mover tape device in a 3494 robot:

```
% /opt/hpss/bin/GetESANumbers <device>
```

- Identify range(s) of tape cartridge labels to be used by HPSS.
- Configure the **lmcp** daemon on the node that will run the HPSS 3494 PVR, verify that it is working properly, and configure the lmcp daemon to be started automatically during system reboot.

To see if **lmcp** daemon is running:

```
% ps -e | grep lmcpd
```

To start the **lmcp** daemon:

```
root% /etc/methods/startatl
```

To test whether **lmcp** daemon is configured and working correctly:

```
% mtlib -l<lmcpDevice> -qL
```

where **lmcpDevice** is usually **/dev/lmcp0**.

To test ability to use **lmcp** daemon to mount a tape:

```
% mtlib -l/dev/lmcp0 -m -V<tapeLabel> -x<deviceNumber>
```

Test ability to dismount the tape:

```
% mtlib -l/dev/lmcp0 -d -V<tapeLabel> -x<deviceNumber>
```

To automatically start the **lmcp** daemon after system reboot, add **/etc/methods/startatl** to the **/etc/inittab** file

Refer to Section 5.2.8.2: *3494 PVR Specific Configuration* of the *HPSS Management Guide* for more information.

4.3.4. STK

For an STK tape library:

- If using an STK tape library, configure the ACSLS and SSI software properly and verify that it is working correctly.

To test the ability to mount and dismount a tape in an STK library, use the **stk_ctl** utility.

To mount a tape:

```
% stk_ctl mount <driveSpec> <tapeLabel>
```

where **driveSpec** is four integers separated by commas (no spaces), identifying the ACS, LSM, panel, and drive (e.g., 0,0,1,2).

To dismount a tape:

```
% stk_ctl dismount <driveSpec>
```

To query a drive:

```
% stk_ctl query <driveSpec>
```

Refer to Section 5.2.8.6.: *STK PVR Specific Configuration* and Section 5.2.8.7: *STK RAIT PVR Configuration* (both in the *HPSS Management Guide*) for more information.

4.3.5. AML



The AML PVR is supported by special bid only.

For AML tape libraries:

- If using an AML PVR, configure the Insert/Eject ports using the configuration files `/var/hpss/etc/AML_EjectPort.conf` and `/var/hpss/etc/AML_InsertPort.conf`.

Refer to Section 5.2.8.3: *AML PVR Specific Configuration* of the *HPSS Management Guide* for more information.

4.4. Verify Tape Drives

Verify that the correct number and type of tape devices are available on each Tape Mover node.

4.4.1. AIX

The tape devices section of the **lsnode** report displays all available tape drives.

To determine the type and number of available tape devices:

```
% lsdev -C -S a -c tape
```

- On each Tape Mover node, verify that each tape drive has the variable-length block size option enabled. The tape devices section of the **lsnode** report will indicate the status of each drive.

To determine if the variable block size option is enabled, the following should return a value of zero(0):

```
% lsattr -E -l <tapeDevice> -a block_size -F value
```

To change the device to use variable block size:

```
root% chdev -l <tapeDevice> -a block_size=0
```

- If using STK drives, verify that the drive type is not incorrectly set to Generic tape drive or IBM Emulation mode.
- On each Tape Mover node, verify that the raw read and write I/O performance of all HPSS tape drives are at expected levels. Create one or more tables documenting the results.

To conduct the read and write tests on **rmt1**, mount a scratch tape on **rmt1** and issue the following commands.



WARNING: The contents of this tape will be overwritten by ioccheck so be sure to mount the correct tape cartridge.

To measure uncompressed write performance (Note that specifying **rmt1.1** causes the tape not to rewind):

```
% ioccheck -w -t 20 -b 1mb /dev/rmt1.1
```

To measure the maximum-compressed write performance on **rmt1** (and then rewind the tape):

```
% ioccheck -w -t 20 -f 0 -b 1mb /dev/rmt1
```

To measure read performance on drive **rmt1** using the previously-written uncompressed and compressed files:

```
% iocheck -r -t 20 -b 1mb /dev/rmt1.1  
% iocheck -r -t 20 -b 1mb /dev/rmt1.1
```

To unload a tape:

```
% tctl -f <device> rewoffl
```

Repeat the above steps for each tape drive.

4.4.2. Solaris

On each Tape Mover node, verify that each tape drive has the variable-length block size option enabled.

To determine if the variable block size option is enabled, the following should complete successfully:

```
% dd if=/dev/null of=/dev/rmt/0n bs=80 count=1  
% dd if=/dev/null of=/dev/rmt/0n bs=1024 count=1
```

If the variable-length block size option is not enabled, consult your driver documentation for procedures to enable it.

On each Tape Mover node, verify that the raw read and write I/O performance of all HPSS tape drives are at the expected levels. Create one or more tables documenting the results.

To conduct the read and write tests on /dev/rmt/0, mount a scratch tape on /dev/rmt/0 and issue the following commands.



WARNING: The contents of this tape will be overwritten by iocheck so be sure to mount the correct tape cartridge.

To measure uncompressed write performance (Note that specifying 0n will cause the tape not to rewind):

```
% iocheck -w -t 20 -b 1mb /dev/rmt/0n
```

To measure the maximum-compressed write performance on 0 and then rewind the tape:

```
% iocheck -w -t 20 -f 0 -b 1mb /dev/rmt/0
```

To measure read performance on drive 0 using the previously-written uncompressed and compressed files:

```
% iocheck -r -t 20 -b 1mb /dev/rmt/0n  
% iocheck -r -t 20 -b 1mb /dev/rmt/0n
```

To empty the tape:

```
% mt -f /dev/rmt/0n rewind  
% mt -f /dev/rmt/0n weof 2
```

4.4.3. IRIX

On each Tape Mover node, verify that each tape drive has the variable-length block size option enabled.

To determine if the variable block size option is enabled, the following should complete successfully:

```
% dd if=/dev/null of=/dev/rmt/tps2d6nr bs=80 count=1
% dd if=/dev/null of=/dev/rmt/tps2d6nr bs=1024 count=1
```

If the variable-length block size option is not enabled, consult your driver documentation for procedures to enable it.

On each Tape Mover node, verify that the raw read and write I/O performance of all HPSS tape drives are at the expected levels. Create one or more tables documenting the results.

To conduct the read and write tests on **tps2d6**, mount a scratch tape on **tps2d6** and issue the following commands.



WARNING: The contents of this tape will be overwritten by ioccheck so be sure to mount the correct tape cartridge.

To measure uncompressed write performance (Note that specifying **tps2d6nr** will cause the tape not to rewind):

```
% ioccheck -w -t 20 -b 1mb /dev/rmt/tps2d6nr
```

To measure the maximum-compressed write performance on **tps2d6** (and then rewind the tape):

```
% ioccheck -w -t 20 -f 0 -b 1mb /dev/rmt/tps2d6nrc
```

To measure read performance on drive **tps2d6** using the previously-written uncompressed and compressed files:

```
% ioccheck -r -t 20 -b 1mb /dev/rmt/tps2d6nr
% ioccheck -r -t 20 -b 1mb /dev/rmt/tps2d6nr
```

To empty the tape:

```
% mt -f /dev/rmt/tps2d6 rewind
% mt -f /dev/rmt/tps2d6 weof 2
```

4.4.4. Linux

There are currently no known tools available to measure raw read and write I/O performance of HPSS tape drives for Linux.

4.5. Setup Disk Drives

Verify that Ultra SCSI is enabled for all SCSI devices via **smit** on AIX and by the appropriate means on non-AIX platforms.

4.5.1. AIX

- Verify that the correct number and type of disk devices are available on each DB2 and Disk Mover node.

The disk devices section of the **lsnode** report displays all available disk devices.

To determine the type and number of available disk devices:

```
% lsdev -C -S a -c disk
```

- If using SSA disks spread the SSA disks equally across the two loops on each SSA adapter.

The SSA configuration section of the **lsnode** report provides details on the SSA configuration.

Note the following:

- The section shows which SSA adapter to which each disk is attached.
 - There are two loops (a and b) per adapter and two ports per loop (a1, a2, b1, b2)
 - The physical order of the disks are shown from the perspective of each port
 - A disk is accessed according to its closest port (e.g., either a1 or a2, b1 or b2)
 - When configuring striped SSA disks in HPSS, it is important to select disks for each striped virtual volume that span ports, loops, and/or adapters. These decisions can be made after determining the probable bottlenecks and then selecting individual disks in a virtual volume to alleviate bottlenecks in the port, loop, or adapter, as desired.
 - For SSA disks on an AIX SP node, use **maymap** to identify which loop the SSA disk is on.
- Create volume groups for all disks to be used by HPSS.
 - Create all necessary raw disk logical volumes to be used by the HPSS Disk Mover(s).

To create a volume group for a physical disk, use **smit** or the following:

```
% mkvg -f -y<volumeGroup> -s<partitionSize> <physicalDisk>
```

To create a logical volume, use **smit** or the following:

```
% mklv -y<logicalVolume> -traw <volumeGroup> <numPartitions>
```

Note that there are additional options for specifying exactly where on the physical disk the logical volume should be placed.

To view all physical disks and associated volume groups:

```
% lspv
```

To view all logical volumes on a given volume group:

```
% lsvg -l <volumeGroup>
```

- On each Disk Mover node, measure the raw read and write I/O performance of all HPSS disks and verify that they are at expected levels. Create one or more tables documenting the results. The output of these tests should be stored in **/var/hpss/stats** for later analysis.

Use the **iocheck.ksh** script from the deployment tools package to show the performance of one or more individual disk devices and the peak aggregate performance of concurrent I/O across multiple disks (e.g., to show the peak performance of adapters).



WARNING: The contents of this logical volume will be overwritten by iocheck so be sure to use the correct logical volume name.

To measure the individual and aggregate throughput of hdisks 4, 5, 6, and 7:

```
% iocheck.ksh 4 5 6 7
```

To measure read performance on a single disk:

```
% iocheck -r -t 20 -b 1mb /dev/r<logicalVolume>
```

where **logicalVolume** is a raw logical volume that is sized to provide at least 20 seconds of I/O throughput.

To measure write performance on a single disk:

```
% iocheck -w -t 20 -b 1mb -o 1mb /dev/r<logicalVolume>
```

where **logicalVolume** is a raw logical volume that is sized to provide at least 20 seconds of I/O throughput.

4.5.2. Linux

Verify that there is a raw device mapping for each block device to be accessed by HPSS. To list all current device mappings:

```
% raw -a -q  
  /dev/raw/raw1: bound to major 8, minor 1  
  /dev/raw/raw2: bound to major 8, minor 2
```

Block devices can be mapped to raw devices at boot time by adding mapping information to the **/etc/sysconfig/rawdevices** file. To map the first partition of SCSI disk **a** to raw device **raw1** and the second partition of SCSI disk **b** to raw device **raw2**, add the following lines:

```
/dev/raw/raw1 /dev/sda1  
/dev/raw/raw2 /dev/sdb2
```

4.5.3. IRIX

For Solaris and IRIX platforms, specific commands and syntax are not listed in this document. Perform the following steps using the appropriate operating system commands:

- Verify that the correct number and type of disk devices are available on each DB2 and Disk Mover node.
- Create all necessary raw disk volumes to be used by the HPSS Disk Mover(s).
- On each Disk Mover node, measure the raw read and write I/O performance of all HPSS disks and verify that they are at expected levels. Create one or more tables documenting the results. The output of these test should be stored in **/var/hpss/stats** for later analysis.

4.6. Setup Network Parameters

- Install and configure all network interfaces and corresponding network connections.

Refer to IBM's internal network technologies home page for resources on configuring and tuning networks and TCP/IP.

The network interfaces section of the **lsnode** report from each node shows the network interfaces that are configured.

To determine how many network interfaces are available on (AIX):

```
% lsdev -C -S a -c if
% netstat -i
```

To determine how many network interfaces are available (Linux):

```
% netstat -i
```

To determine if an SP switch is available (AIX):

```
% lsdev -C -S a -l css0
% netstat -i
```

To view all IP addresses associated with a local host on AIX (i.e., for all network interfaces):
Note: Does not work for SP switch interface.

```
% for x in `lsdev -C -S a -c if -F name`; do
  lsattr -E -a netaddr -l $x -F value
done
```

To view additional information for each network interface (AIX):

```
% for x in `lsdev -C -S a -c if -F name`; do
  ifconfig $x
  lsattr -El $x
done
```

For non-AIX platforms, use the Name column from **netstat -i** in the **ifconfig** commands. Note the IP address follows the inet phrase in the output from the **ifconfig** command.

To test whether an IP address is reachable (non-zero exit status indicates the **ping** was not successful):

```
% ping -c 1 <ipAddress>
```

- Isolate Kerberos communication to the designated control network(s) on all HPSS and DB2 nodes in order to separate the HPSS control and data paths.
- Place all HPSS, DB2, and Kerberos server node IP addresses in a local host table (**/etc/hosts**). For AIX, configure the node to use the table as backup in the event of a DNS failure. The file **/etc/netsvc.conf** should be modified to look like the following:

```
hosts=bind,local
```

The **netsvc.conf** file is used to specify the ordering of host name resolution. In the above ordering, DNS will be used first; if the host name is not found, then the local **/etc/host** file

will be used.

For Linux, a similar change should be made to `/etc/nsswitch.conf`:

```
hosts:      nis dns files
```

- For each AIX ethernet network interface, verify that the `en0` and `et0` interfaces are not both configured at the same time (we recommend only using `en0` unless the other nodes in the network are all using the **802.3 et*** interface). Configure the local name service with the unique hostname for each network interface on all nodes and verify that each hostname is resolvable from other nodes.
- Verify that network TCP throughput has been optimized and the performance of each network is at expected levels in both directions (especially check HPSS data networks between Movers and between Mover and client nodes). Using `ttcp` or another network tool, measure the performance of all the networks that will be used in communications between Kerberos, HPSS, DB2, and client nodes. If multiple paths between nodes are to be used, then all of them need to be measured as well. The transfer rates for networks differ greatly depending upon the individual technology and network settings used. It is important to gather performance data using a variety of settings to determine the optimal combinations. The primary values that govern performance include send/receive buffers, size of reads/writes, and `rfc1323` value for high performance networks (HIPPI, G-Enet). Create a table showing these values. An example table can be found below:

To test the receiving node's performance, issue:

```
% ttcp -r -s -b<bsize> -l<lsize>
```

To test the sending node's performance, issue:

```
% ttcp -t -s -b<bsize> -l<lsize> <hostname/interfacename>
```

The following is an example of the information to be gathered from the above commands (assumes `en0` is 10 MB Ethernet, if `en0` is Fast or G-Enet, `rfc1323` should be on):

Receiver Table

Interface	Bsize	Lsize	RFC1323	Performance	
CPU utilization					
en0	16k	16k	Off	--	--
en0	16k	8k	Off	--	--
en0	64k	64k	Off	--	--
en0	64k	32k	Off	--	--
...					
ccs0	64k	64k	On	--	--
ccs0	64k	32k	On	--	--
...					

Sender Table

Interface	Bsize	Lsize	RFC1323	Performance	
CPU utilization					
en0	16k	16k	Off	--	--
en0	16k	8k	Off	--	--
en0	64k	64k	Off	--	--
en0	64k	32k	Off	--	--

```

...
ccs0          64k          64k          On          --          --
ccs0          64k          32k          On          --          --
...

```

You are looking for the best values possible for each network connection. These values will be used by HPSS to optimize its data transfers. This example is, by no means, a complete picture of what controls network performance. In fact, it is assumed that you have already optimized the networks. The reason for gathering these values is to optimize HPSS performance on an already tuned network, not to fix underlying network problems.

To test the TCP socket performance over a network connection, issue the following on the receiving node:

```
% ttcp -r -s -p<port>
```

where a typical port is 4321. Then issue the following on the transmitting node:

```
% ttcp -t -s -p<port> <hostname>
```

Note that the **ttcp** tool is included in the deployment package and is not related to the UNIX ToolTalk service.

HPSS makes extensive use of a system's networking capabilities. Therefore, the setting of the tunable networking parameters for the systems on which the various HPSS servers and clients will run can have a significant impact on overall system performance.

AIX provides a utility program to display and modify a number of networking parameters. The utility is **no** (Network Options). Refer to the *AIX Performance Tuning Guide*, for details of each option and its impact on networking and system performance.

Some options that typically impact performance within an HPSS system environment are:

Table 8. Network Options

Network Option	Description
thewall	Controls the maximum amount of system memory that can be used by the system networking code. A value that is too low can cause networking requests to be delayed or denied. The recommendation from AIX development is to set this value to at least two times the maximum number of concurrent connections times the size of the socket send/receive buffers. The default setting for AIX 4.3.2 and later is the smaller of (1) half the size of physical memory or (2) 1 GB.
sb_max	Controls the maximum size allowed for send and receive buffers for a socket.

udp_recvspace	Controls the default size of the receive buffer for UPD/IP sockets. A value that is too small can cause server RPC sockets to be overrun.
tcp_recvspace, tcp_sendspace	Controls the default size for the receive and send buffers for TCP/IP sockets. Internally, HPSS servers and clients attempt to set these buffers sizes explicitly, but other utilities may not.
rfc1323	Controls whether large TCP window sizes are used. Usually set to ON for higher throughput networks (e.g., HiPPI, SP switch) and set to OFF for lower throughput networks (e.g., ethernet, FDDI).

AIX also provides a configuration attribute that controls the maximum amount of memory that can be allocated to **mbufs**. It can be viewed or modified via **smit** (select “**Process Environments**”, then “**Change / Show Characteristics of Operating System**”) or via the command line (“**lsattr -E -l sys0**”, “**chdev -e sys0 -a maxmbuf = <new value>**”).

It is recommended that the available combination of options be tested as part of the initial HPSS system testing. In particular, poor network performance has been experienced where options on one system do not match options on other remote systems.

There are also attributes that are specific to the individual network interface that may affect network performance. For example, the network interface for the IBM SP TB3 switch provides settings for the size of the send and receive pool buffer size, which have had an effect on throughput. It is recommended that the available interface specific documentation be referenced for more detailed information.

The anticipated load should also be taken into account when determining the appropriate network option settings. Options that provide optimal performance for one or a small number of transfers may not be the best settings for the final multi-user workload.

4.6.1. HPSS.conf Configuration File

The **HPSS.conf** configuration file contains tuning options to be used by HPSS clients and servers.

The **HPSS.conf** file may also contain options used by non-HPSS applications. Application developers are asked to please observe the “Reserved for Future Use” components specified in Appendix D.

The **HPSS.conf** configuration file is located in the directory named by either:

- The **HPSS_CFG_FILE_PATH** environment variable,
- the directory **/usr/local/etc**
- the directory **/var/hpss/etc** (Preferred), or
- in that order. If the file is not present or no matching entry is found, the Parallel FTP Client, Client API, and Mover will use system defaults.

See Appendix D: *HPSS.conf Configuration File* for more details.

4.6.2. SP/x Switch Device Buffer Driver Buffer Pools

IBM SP/x systems provide the capability to tune the buffer pool allocation in the switch device driver. Two variables can be changed: `rpoolsize`, which is the size of the buffer pool for incoming data, and `spoolsize` which is the buffer pool size for outgoing data. If these values are too small, then buffer overruns may occur.

The current values of these variables can be interrogated with the `lsattr` command (e.g., `lsattr -E -l css0`), and can be changed with the `chgcsm` command (e.g., `chgcsm -l css0 -a spoolsize=<new size> -a rpoolsize=<new size>`). Refer to the IBM *Parallel System Support Programs for AIX Administration Guide*, GC23-3897-02 for more details.

Chapter 5. HPSS Installation and Infrastructure Configuration

This chapter provides instructions and supporting information for installing the HPSS prerequisite software, the HPSS and DB2 software from the HPSS distribution media, and performing the HPSS infrastructure configuration.

To install and setup an HPSS system, we recommend that the administrator be familiar with UNIX commands and configuration, be familiar with a UNIX text editor, and have some experience with shell scripts.



Note: For information on upgrading from a previous version of HPSS, please see Chapter 6: Upgrading to HPSS Release 6.2 .

The steps required to install and setup an HPSS system are listed below. Each step is discussed in more detail in the section referenced.

- Prepare for Installation (Section 5.1)
- Install Prerequisite Software (Section 5.2)
- Install HPSS/DB2 and Configure HPSS Infrastructure (Section 5.3)
- Post Installation Procedures (Section 5.4)
- HPSS Documentation & Manual Page Setup (Section 5.5)
- Define HPSS Environmental Variables (Section 5.6)
- Tune DB2 (Section 5.7)
- Install and Build HPSS Source Code (Section 5.8)

5.1. Prepare for Installation

The following sections discuss the steps that need to be taken in order to prepare the system for HPSS installation.

5.1.1. Distribution Media

Obtain the HPSS software from your IBM HPSS customer support representative. The software is available on one of the following distribution media:

- CD-ROM (available 6.2.1 or later)
- Image files

5.1.2. Software Installation Packages

The HPSS software is provided in the following packages:

- Server - Contains all HPSS binaries and libraries
- Client/Mover - Contains HPSS file necessary to run an HPSS client and the HPSS Mover binaries.
- Source – Contains HPSS source files. This package is available only by special arrangement.
- SSM Help – Contains the HPSS Management Guide in HTML format which is used by the SSM's Help menu to display window-specific help topics.
- The HPSS software package names and sizes for the supported platforms are as follows:

Table 9. Installation Package Sizes and Disk Requirements

Platform	HPSS Package Name	Package Size	/opt/hpss Space Requirements	Package Description
AIX	HPSSServer-6.2.0.0.lpp	< 65MB	450 MB	All HPSS Components
	HPSSClientMover-6.2.0.0.lpp	< 10MB	70 MB	Client and Mover Components
	HPSSSource-6.2.0.0.lpp	< 40MB	130 MB	HPSS Source Code
Linux	HPSSServer-6.2.0.0.rpm	< 20 MB	175 MB	All HPSS Components
	HPSSClientMover-6.2.0.0.rpm	< 5 MB	70 MB	Client and Mover Components
	HPSSSource-6.2.0.0.rpm	< 10 MB	130 MB	HPSS Source Code
IRIX	HPSSClientMover-6.2.0.0.tar.Z	< 5 MB	75 MB	Client and Mover Components
SUN	HPSSClientMover-6.2.0.0.pkg	< 10 MB	75 MB	Client and Mover Components
(ALL)	SSMHelp.tar	< 15MB	25 MB	SSM Help Files

5.1.3. Create Owner Account for HPSS Files

The HPSS software must be installed by a **root** user. In addition, a UNIX User ID of **hpss** and Group ID of **hpss** is required for the HPSS installation process to assign the appropriate ownership for the HPSS files. If the hpss User ID does not exist, the installation process will create it with the **mkhps** tool based on default UID/GID values defined by the `~hpss/include/hpss_env_def.h` file. If alternate IDs are required, this include file should be updated prior to (re)building **mkhps**, or the 'hpss' account/group should be created beforehand using standard administrative procedures. Also note that HPSS will need a number of other user/group IDs for various servers and prerequisite components. These IDs are also pre-defined in the same include file and can be modified to support a site's particular account policy.



It is very important that the HPSS file permissions be set up properly. If they are not, HPSS may not work properly after it is configured. We recommend that the HPSS file ownerships and permissions set by the installation process be preserved. If they must be changed, care must be taken to ensure they are changed correctly. Refer to Section 5.4: Post Installation Procedures on page 169 for more information on the HPSS file ownerships and permissions.

5.1.4. Installation Target Directory Preparation

By default, the HPSS software is installed in the `/opt/hpss` directory. Before installing the HPSS software, make sure that the installation target directory satisfies the following conditions:

- The `/opt/hpss` directory is not being used.

- The disk, where the installation target directory resides, has enough space to hold all the HPSS packages to be installed on this node.
- **WARNING:** Do not use NFS mounted directories for installing nor allocating space for HPSS related components. Installing on NFS is problematic and the errors can be difficult to diagnose.

5.2. Install Prerequisite Software

This section provides an overview of how to install the prerequisite software to prepare for the upcoming HPSS configuration. Verify that the correct software versions are obtained as described in Section 3.3.1: *Prerequisite Software Overview* on page 58.

5.2.1. Install Java

Java is required to compile HPSS software. It is also required to run the SSM Graphical User Interface (hpssgui), the SSM Command Line Interface (hpssadm) and the **mkhpss** tool. If the site needs to compile the HPSS code, the Java Software Development Kit (SDK) must be installed. To run HPSS applications, either the Java SDK or the Java Runtime Environment (JRE) is required.

It is recommended that the Java SDK component be installed on the machines where HPSS code is to be compiled and on the machine where **mkhpss** will be running. Also, the JRE component should be installed on each SSM client machine. This is to enable the **hpssgui** application to run directly on the client machine to maximize hpssgui performance and to eliminate the X traffic on the HPSS server machine.

See Section 3.3.1.4: *Java* on page 59 to obtain the download website. Follow the installation instructions provided on the website to install Java. Be sure that all Java's prerequisite requirements are met before installing the product.

5.2.2. Install MIT Kerberos (If Using Kerberos Authentication)



The capability to use MIT Kerberos authentication is provided in HPSS 6.2, however, IBM Service Agreements for HPSS do not provide support for defects in MIT Kerberos. Kerberos maintenance/support must be site-provided.

For Linux, Kerberos is included in the Operating System and does not need to be installed.

For AIX, download the Kerberos 1.4.2 source from <http://web.mit.edu/kerberos>. Extract the source tree into the desired location; the recommended location is /usr/local. Under the newly extracted tree go to krb5-1.4.2/source. Execute the following commands:

```
> configure --enable-shared
> make
> make install
```

MIT Kerberos will be installed in the configured directory.

5.2.3. Install LDAP (If Using LDAP Authorization)



LDAP authorization is not supported by IBM Service Agreements. The following information is provided for sites planning to use LDAP authorization with HPSS 6.2 as a site supported feature.



If UNIX authorization will be used, this product is not required. LDAP authorization is supported on AIX only.

If LDAP will be used for HPSS authorization, download the IBM Tivoli Directory Server from <http://www.ibm.com>. Search for "IBM Tivoli Directory Server" on the site to find the download page. The download page presents many options for downloading the software. Download version 6.0 of the package that is a tar file containing the LDAP software and its prerequisites. Find a temporary directory with plenty of space and extract the tar file. There will be a script named "setup" in the root directory where the tar file was extracted. Run this script to start an installation wizard to install the software. When the installation program offers to install DB2, do not install it as it is a trial version. LDAP will use the version of DB2 that is installed from the HPSS installation CD.

Installing the IBM Kerberos client is required only if LDAP is being used for authorization and you will be using Kerberos authentication with the LDAP daemon. The fileset for the IBM Kerberos client is located on the AIX 5.2 Expansion Pack CD #2. Use the normal AIX software installation programs (e.g. `installp` or `smit`) to install the `krb5.client.rte` fileset from that CD. If you will be using Kerberos authentication with LDAP, the `LIBPATH` environment variable must be set to include `/usr/krb5/lib`. Since this is required for all clients and servers, this should be set in `/etc/environment`. Because HPSS requires the use of MIT Kerberos, the `LIBPATH` must also contain the path to the `lib` directory of the MIT Kerberos installation. For example, if `/etc/environment` already contains a setting "`LIBPATH=/usr/lib`", and MIT Kerberos is installed under `/usr/local`, the setting in the file `/etc/environment` should be changed to "`LIBPATH=/usr/lib:/usr/local/lib:/usr/krb5/lib`".



If planning to use the LDAP option, the default binaries are not enabled to communicate with the LDAP system. HPSS must be recompiled/relinked. The HPSS source images/packages must be installed and the `Makefile.macros` file updated to "turn on" LDAP. Look for the line with `LDAP_AUTH_SUPPORT`, and change the "off" to "on". Rebuild HPSS according to the instructions in Section 5.8.1.2: Build the HPSS Base Source Tree on page 172.

5.2.4. Install Prerequisite Software for XFS HDM



XFS is not supported in HPSS 6.2. XFS references have been left in the HPSS documentation to support the option of re-enabling XFS in future releases.

The XFS HDM is available only on Linux. Unlike the other components of HPSS which run under the Redhat Linux distribution, the XFS HDM requires the SUSE Linux distribution. This is because Redhat kernels do not have support for the Data Management Application Programming Interface (DMAPI). DMAPI is required to link an XFS filesystem to HPSS.

To prepare a machine to run the HPSS HDM:

- Install SUSE LINUX Enterprise Server (SLES) version 9 Service Pack 2 or greater. See <http://www.suse.com> for details.
- Install MIT Kerberos 1.3.5
- Use `yast` to install the following XFS and DMAPI userspace programs:
 - `dmapi-2.2.1-0.2` or higher
 - `xfsdump-2.2.25-0.2` or higher
 - `xfsprogs-2.6.25-0.2` or higher

XFS uses the Linux udev daemon to dynamically configure the DMAPI device node. Use the **chkconfig** command to make sure that boot.udev is "on". Note that if a change is required, then the machine needs to be rebooted in order for it to take effect.

5.3. Install HPSS/DB2 and Configure HPSS Infrastructure

The HPSS installation and infrastructure configuration must be performed on the following HPSS machines:

- Root Subsystem
- Secondary Subsystem
- Mover/Client

Although the installation and configuration steps to be performed are similar, each step will be processed differently depending on the machine type. Therefore, it's very important that the correct submenu (Root Subsystem, Secondary Subsystem, or Mover/Client) is selected to perform the configuration. The following sections describe the procedure to be performed on each machine.

5.3.1. Install and Configure HPSS - Root Subsystem Machine

Perform the following steps on the root subsystem machine:

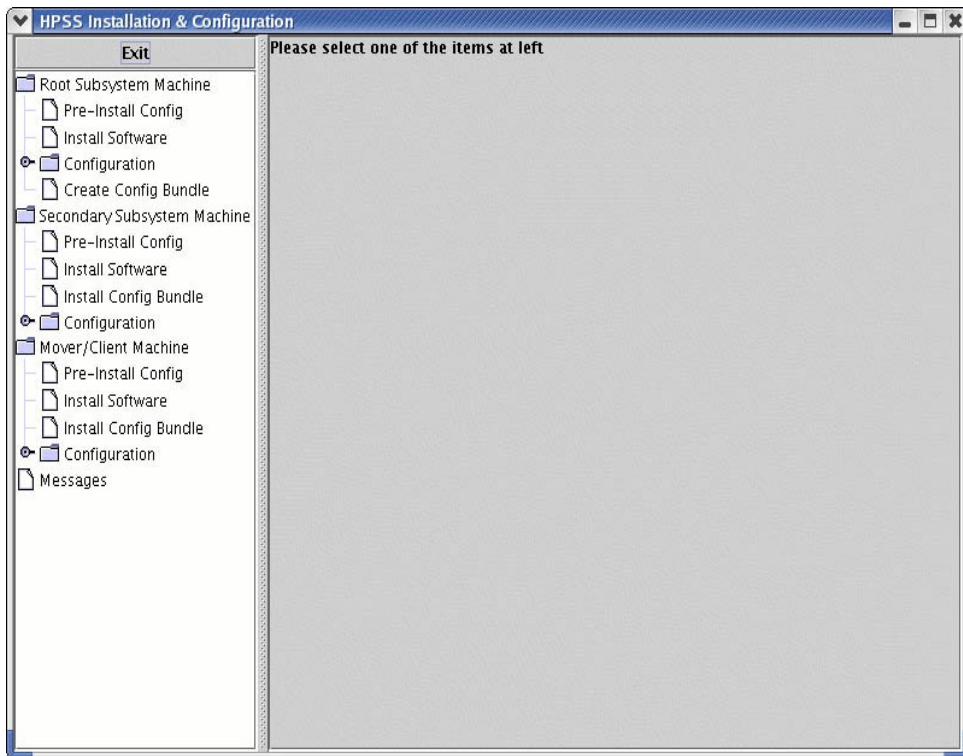
- Pre-installation Configuration
- Install HPSS documentation and DB2
- Set up DB2 permanent license
- Configure Security Services
- Configure DB2
- Configure Other Services
- Create Configuration Bundle

The following sections describe the procedures to perform these steps.

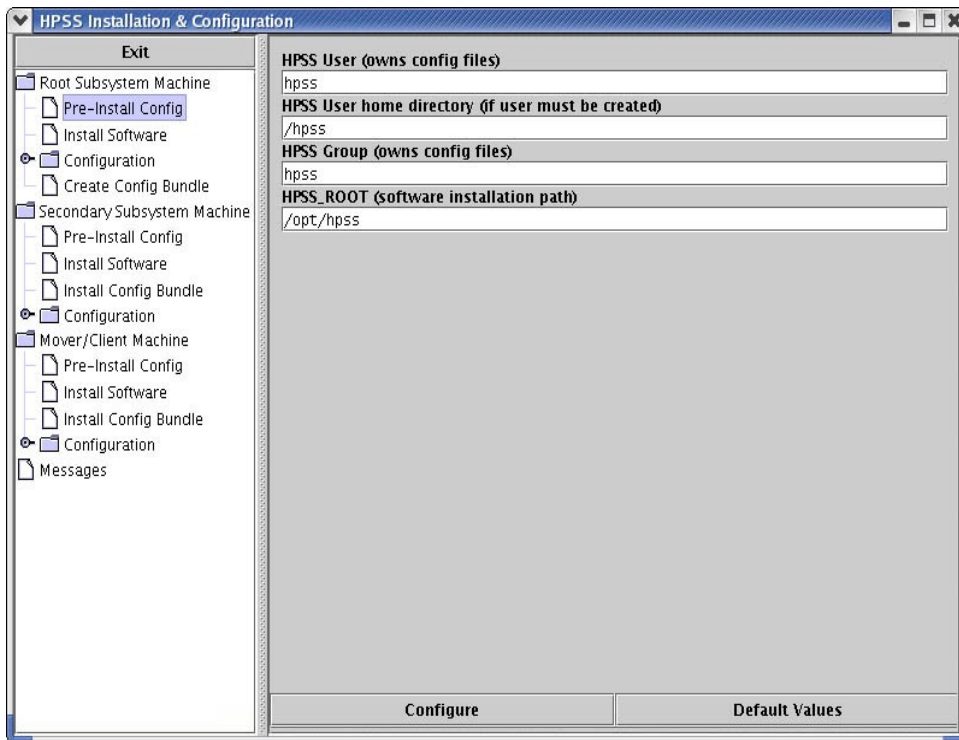
5.3.1.1. Pre-Installation Configuration

Before configuring HPSS, perform the following steps to set up the appropriate configuration directories and files:

4. Install the appropriate install images on the target machine and verify that the **mkhpss** utility is available. On AIX, an installp image or tar file will be provided. On Linux, an RPM package or tar file will be provided. If default directories are used, the **mkhpss** tool should be in the /opt/hpss/bin directory. Invoke the **mkhpss** utility and you should see the following screen.



- From the "Root Subsystem Machine" submenu, click on the 'Pre-Install Config' icon in the left panel. **mkhpss** will display the following screen:



- Verify that the default values are correct for the given installation and modify if necessary. Click the 'Configure' button to perform the pre-installation setup. This will run a set of

scripts to verify/create the 'hpss' account and group, setup the /var/hpss (default location) directory with the required subdirectories and initialize the HPSS environment file, env.conf in /var/hpss/etc.

7. If the pre-installation configuration succeeds, you will see the following message in the command output window:

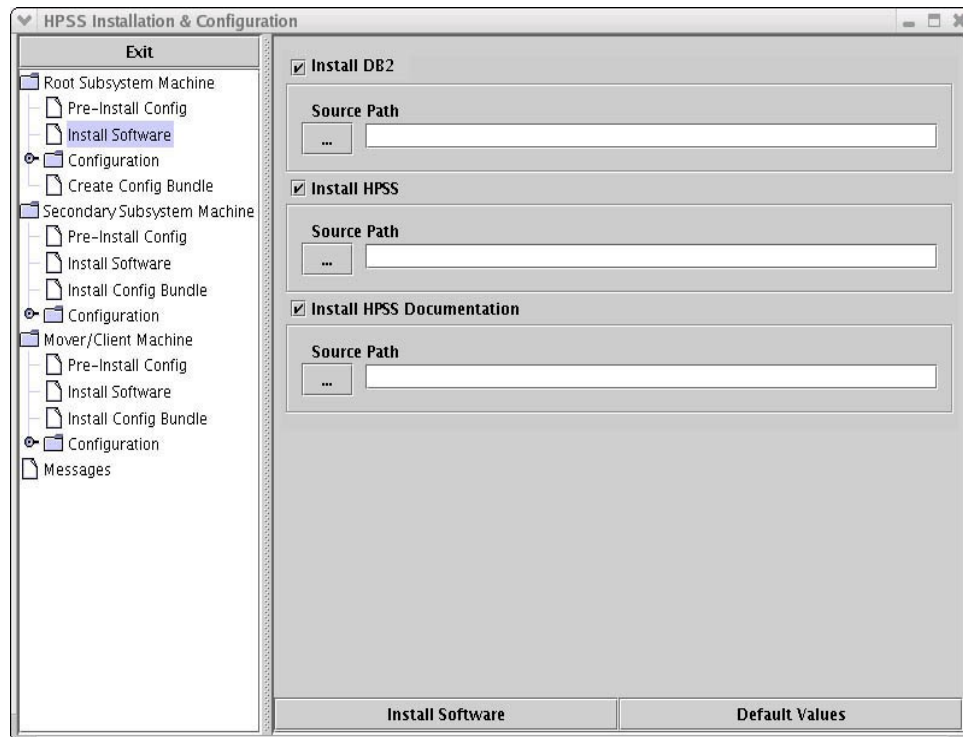
```
## run command exited with status 0
```

Click the 'Done' **button** on the Command Output window to close the window.

5.3.1.2. Install HPSS Documentation and DB2 Software

This section describes the procedure to install HPSS documentation and DB2 software on the root subsystem machine.

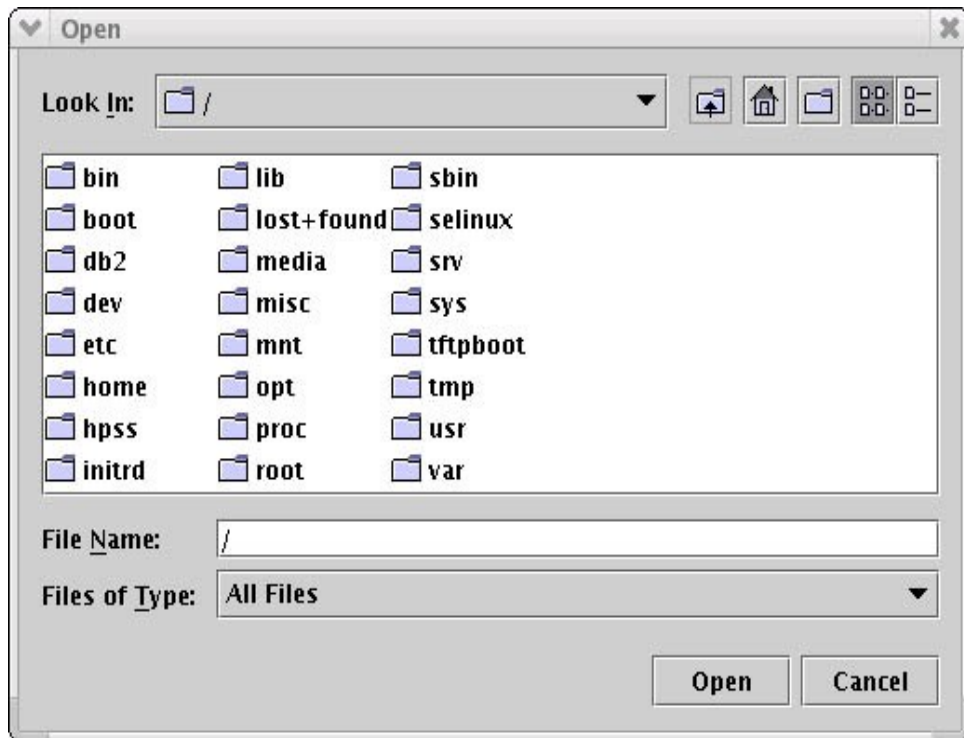
From the "Root Subsystem Machine" submenu in the left panel, click on the "Install Software" option. The right panel will be displayed as shown:



This panel allows you to install DB2 software, HPSS software, and HPSS documentation. Perform the following steps to install the software:

1. To install DB2, check the 'Install DB2' checkbox. Click on the 'Source Path' button and select the directory that contains the DB2 filesets you wish to install.

The "Source Path" button will display the following screen to allow you to specify the desired directory:



DB2 software will be installed in the /usr/opt/db2_08_01 (AIX) or /opt/IBM/db2/v8.1 (Linux) directory.

2. For 6.2.0, the HPSS software and documentation should already be installed according to the installation step in section 5.3.1.1. Installing from CD will be an option starting in 6.2.1.
3. For 6.2.0, the **hpssuser** utility should be used to package the HTML files for delivery to each **hpssgui** machine or to a common shared file system on a remote node. Installing from CD will be an option starting in 6.2.1.



To provide the SSM Help facility, the HTML files from the HPSS documentation must be accessible from the node where the hpssgui runs. The hpssuser utility may be used to package the HTML files for delivery to each hpssgui machine or to a common shared file system. Installation of the HPSS documentation elsewhere is optional.

4. Click the 'Install Software' button to install the selected software/options on the system.
5. If the installation was a success, you should see the following message in the command output window:

```
## run command exited with status 0
```

Click the 'Done' **button** on the Command Output window to close the window.

5.3.1.3. Set Up DB2 Permanent License

This section describes the procedure to create the DB2 permanent license and to specify the number of processors licensed for the root subsystem machine.

To create a permanent DB2 license, issue the following commands:

```
% su -
% cd /usr/opt/db2_08_01/adm
% ./db2licm -a <path name to the DB2 generic license file>
```

The generic DB2 license file (*db2/license/db2ese.lic) can be found on the DB2 Installation CD or image. It can also be obtained by contacting your HPSS Support Representative.

To update the license with the appropriate number of processors, issue the following command:

```
% ./db2licm -n db2ese <number of processors>
```

Refer to the DB2 Command Reference document for more information on how to use the db2licm utility to manage the DB2 license.

5.3.1.4. Configure HPSS Security Services

This section describes the authentication and authorization mechanisms which can be configured to enable HPSS to provide the desired security services. The following authentication and authorization combinations are supported:

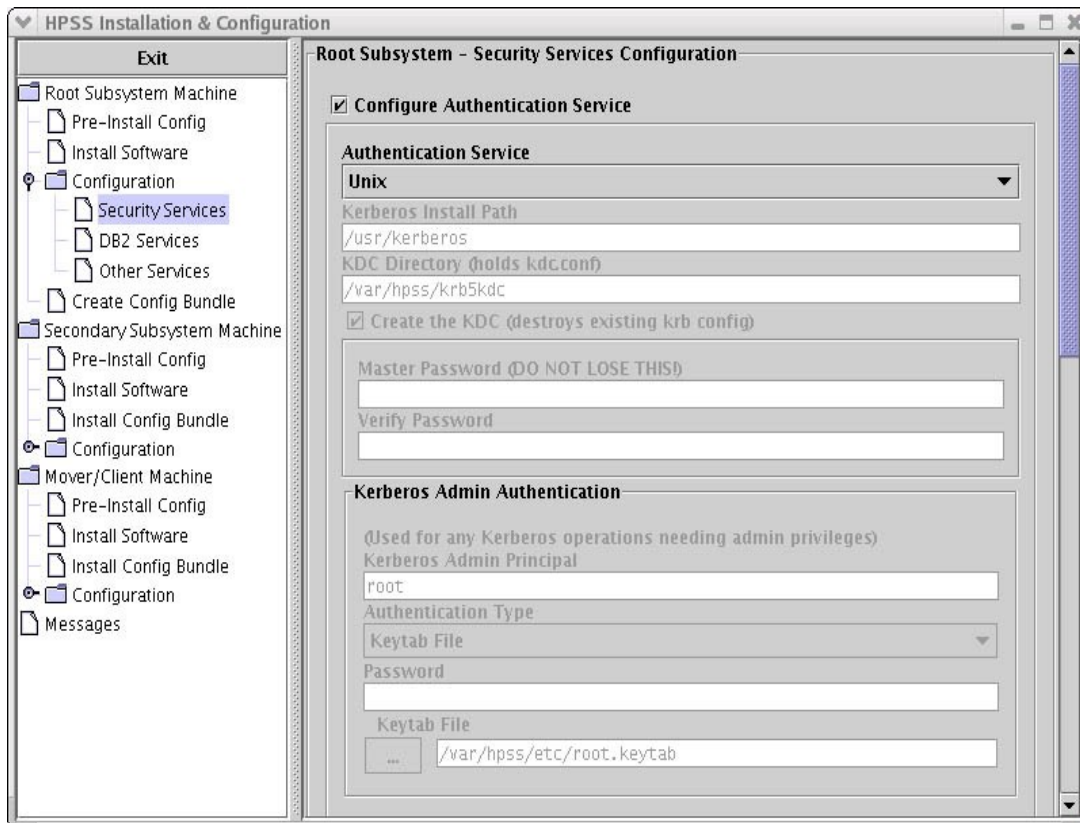
<i>Authentication Mechanism</i>	<i>Authorization Mechanism</i>
UNIX	UNIX
Kerberos	UNIX
Kerberos	LDAP

The following sections describe the procedure to configure the above security combinations.

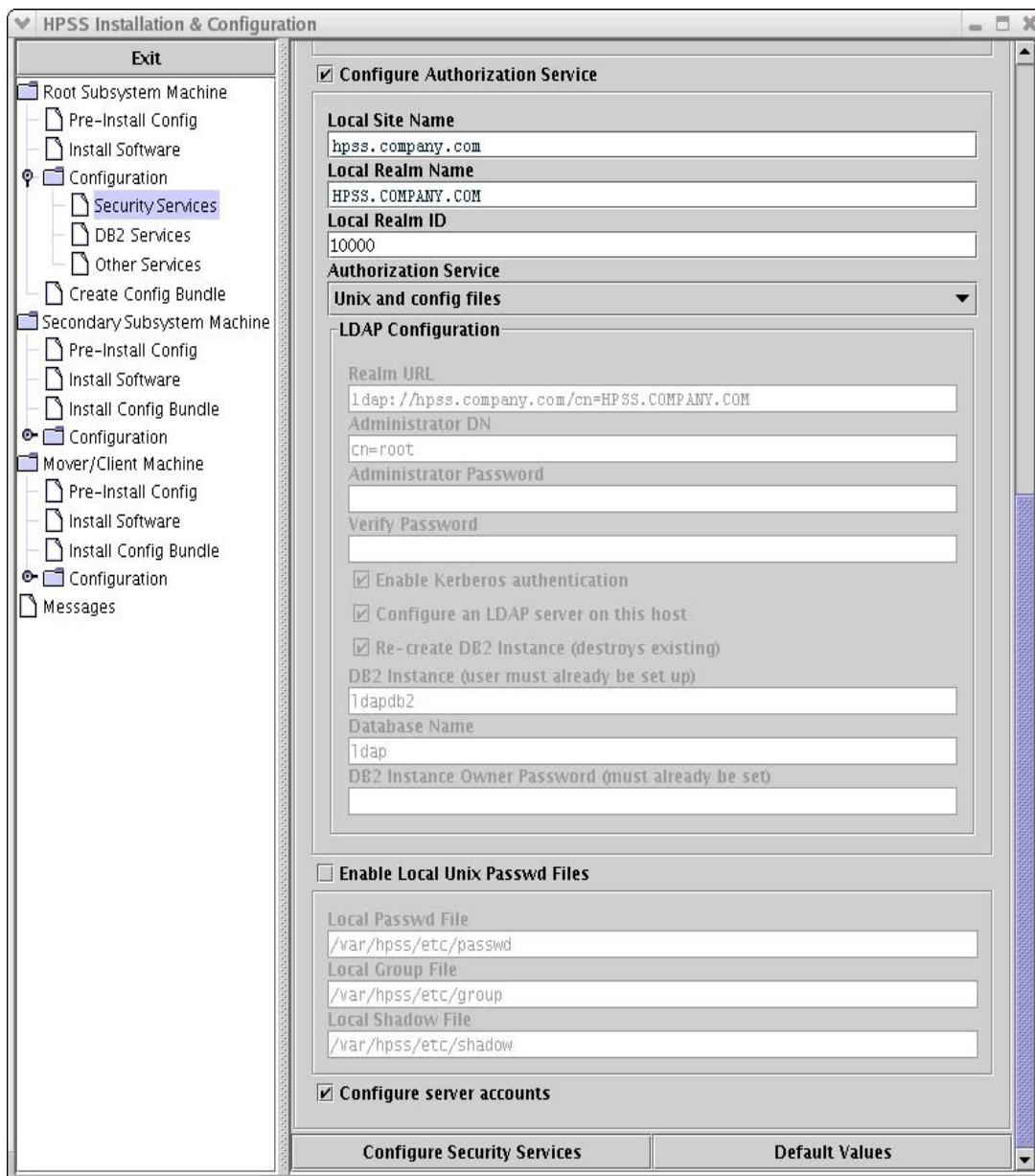
5.3.1.4.1. Configure UNIX Authentication and UNIX Authorization

From the "Root Subsystem Machine" submenu in the left panel, click on the "Configuration" icon and then the "Security Services" icon. Perform the following steps using the right panel :

1. Select the "Configure Authentication Service" checkbox. Set the "Authentication Service" to "Unix".
2. The fields to be set are as displayed on the screen shown:



3. Using the scrollbar, move the right-panel display until the "Authorization Service" information is seen. It should look like the following:



4. Select the "Configure Authorization Service" checkbox. Set the "Authorization Service" to "Unix and config files".
5. Review and modify (if necessary) the active fields:
 - **Local Site Name.** The value is usually set to the full machine name of the local host which can be determined using the 'hostname' and 'domainname' commands.
 - **Local Realm Name.** The value is usually set to the "Local Site Name" all capitalized.
 - **Local Realm ID.** The field is set to a unique ID number for each site. Ask your support representative for your site's value.
6. By default, the system's configuration files (/etc/passwd, /etc/group, and /etc/shadow) are used to administer the authentication and authorization services. As an option, the HPSS configuration files can be used instead. These files are created by **mkhpss** as part of this

configuration step. Other HPSS utilities are available to administer these HPSS configuration files. See section 2.2.2: *Security Mechanisms* in the *HPSS Management Guide* for more information. To use the HPSS configuration files, select the "Enable local Unix Passwd Files" checkbox. The default names for the files should be used as displayed.

7. Select the "Configure Server accounts" checkbox to create UNIX accounts for HPSS servers.
8. Click on the "Configure Security Services" button at the bottom of the screen to perform the specified security configuration.
9. If the configuration was a success, you should see the following message in the command output window:

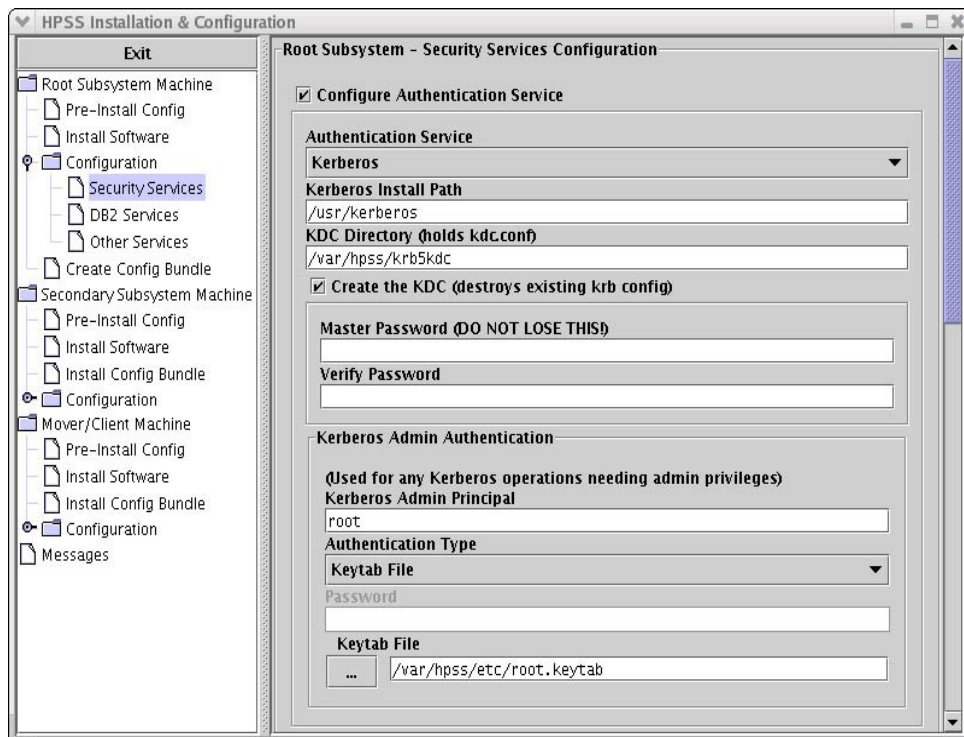
```
## run command exited with status 0
```

Click the 'Done' button on the Command Output window to close the window.

5.3.1.4.2. Configure Kerberos Authentication and UNIX Authorization

From the "Root Subsystem Machine" submenu in the left panel, click on the "Configuration" icon and then the "Security Services" icon. Perform the following steps using the right panel :

1. Select the "Configure Authentication Service" checkbox. Set the "Authentication Service" to "Kerberos".
2. The fields to be configured are as displayed on the screen shown:



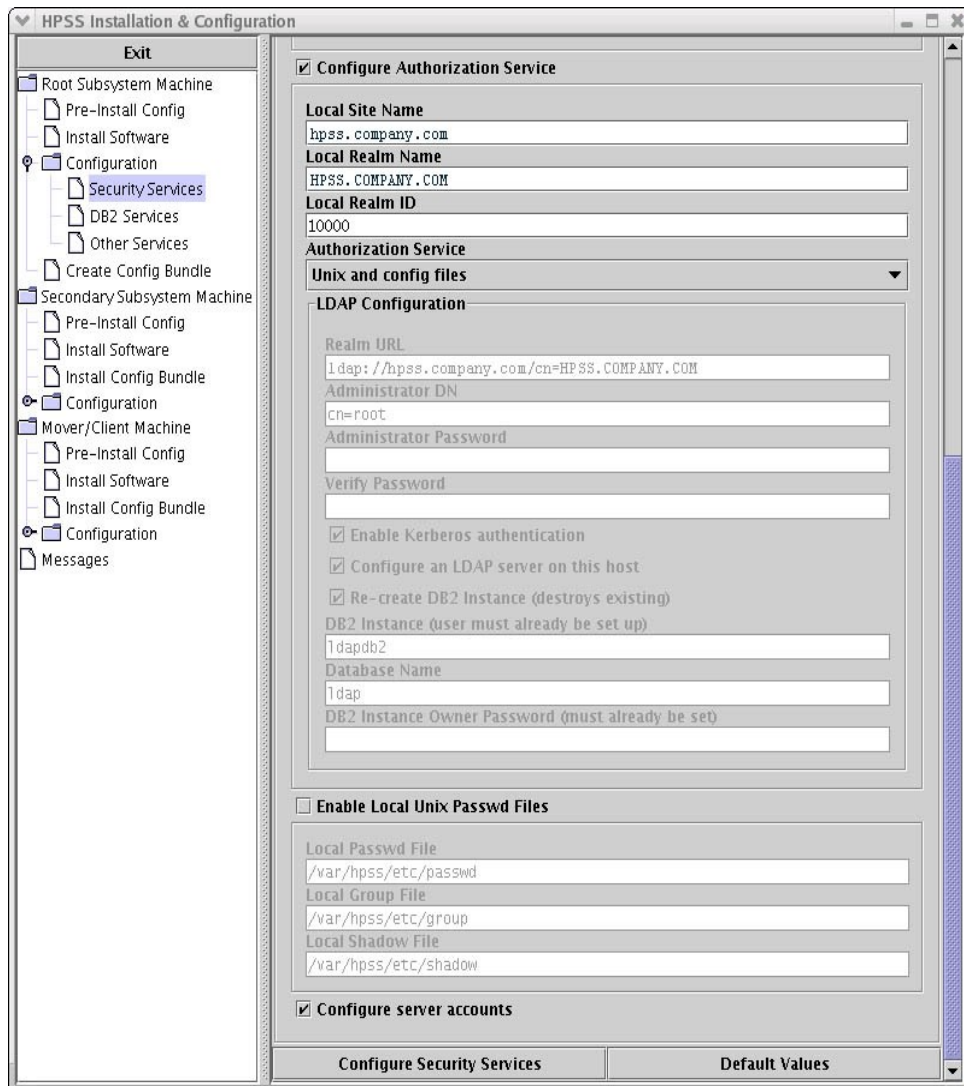
3. Review and modify (if necessary) the following authentication fields:
 - **Kerberos Install Path.** The pathname where Kerberos is installed. The default directory is /usr/kerberos.

- **KDC Directory.** The pathname of the KDC directory. This directory should be set to `/var/hpss/krb5kdc`.
- **Master Password.** The Kerberos administration password.



Be sure to remember this password to be able to administer the Kerberos environment later.

- **Verify Password.** Re-enter the Kerberos administration password.
 - **Kerberos Admin Principal.** The userid to administer the Kerberos environment.
 - **Authentication Type.** There are two supported options: Keytab File or Password. The Keytab File option allows HPSS servers or utilities to read a keytab file to authenticate. The Password option requires a password to be supplied each time an HPSS server or utility is invoked.
 - **Password.** The password used to authenticate the caller when the HPSS server or utility is invoked. This field is not enterable when the Authentication Type field is specified as Keytab File.
 - **Keytab File.** The pathname of the keytab file to be created if the Authentication Type is set to "Keytab File". This file is normally located in the `/var/hpss/etc` directory. This field is not enterable when the Authentication Type field is specified as Password.
4. Using the scrollbar, move the right-panel display until the "Authorization Service" information is seen. It should look like the following:




5. Select the "Configure Authorization Service" checkbox. Set the "Authorization Service" to "Unix and config files".
6. Review and modify (if necessary) the following authorization fields:
 - **Local Site Name.** The value is usually set to the full host name of the local host. This can be determined by using the 'hostname' and 'domainname' commands.
 - **Local Realm Name.** The value is usually set to the "Local Site Name" all capitalized.
 - **Local Realm ID.** The field is set to a unique ID number for each site. Ask your support representative for an appropriate value.
7. By default, the system's configuration files (/etc/passwd, /etc/group, and /etc/shadow) are used to administer the authentication and authorization services. As an option, the HPSS configuration files can be used instead. These files will be created by **mkhpss** as part of this configuration. Other HPSS utilities are available to administer these HPSS configuration files. Refer to Section 2.2.2: *Security Mechanisms* in the *HPSS Management Guide* for more information. To use the HPSS configuration files, select the "Enable local Unix Passwd Files" checkbox. The default names for the files should be used as displayed.

8. Select the "Configure Server accounts" checkbox to create UNIX accounts for HPSS servers.
9. Click on the "Configure Security Services" button at the bottom of the screen to perform the specified security configuration.
10. If the configuration was a success, you should see the following message in the command output window:

```
## run command exited with status 0
```

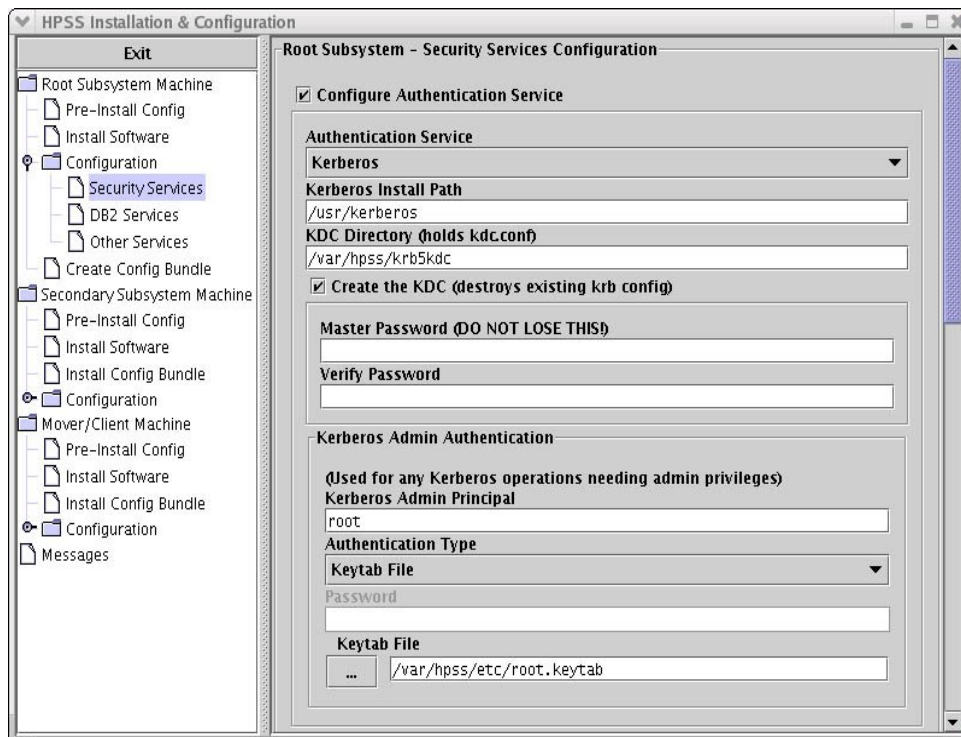
Click the 'Done' **button** on the Command Output window to close the window.

5.3.1.4.3. Configure Kerberos Authentication and LDAP Authorization

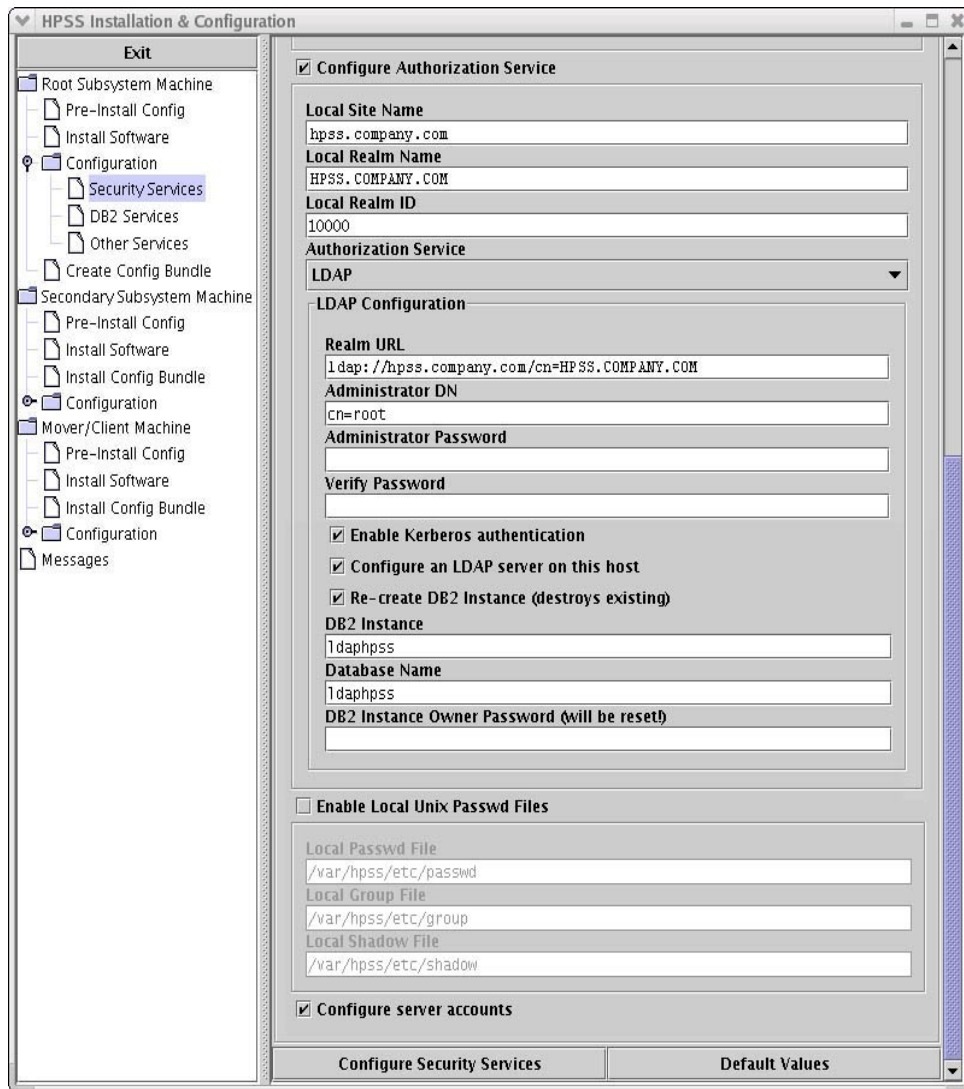
 *LDAP Authorization is supported only on AIX.*

From the "Root Subsystem Machine" submenu in the left panel, click on the "Configuration" icon and then the "Security Services" icon. Perform the following steps using the right panel :

1. Select the "Configure Authentication Service" checkbox. Set the "Authentication Service" to "Kerberos". NOTE: UNIX authentication is not supported when LDAP authorization is selected.
2. The fields to be configured are as displayed on the screen shown:



3. Using the scrollbar, move the right-panel display until the "Authorization Service" information is seen. It should look like the following:



4. Select the "Configure Authorization Service" checkbox. Set the "Authorization Service" to "LDAP".
5. Review and modify (if necessary) the following authentication fields:

- **Kerberos Install Path.** The pathname where Kerberos is installed. The default directory is /usr/kerberos.
- **KDC Directory.** The pathname of the KDC directory. This directory should be set to /var/hpss/krb5kdc.
- **Master Password.** The Kerberos administration password.



Be sure to remember this password to be able to administer the Kerberos environment later.

- **Verify Password.** Re-enter the Kerberos administration password.
- **Kerberos Admin Principal.** The userid to administer the Kerberos environment.

- **Authentication Type.** There are two supported options: Keytab File or Password. The Keytab File option allows HPSS servers or utilities to read a keytab file to authenticate. The Password option requires a password to be supplied each time an HPSS server or utility is invoked.
 - **Password.** The password used to authenticate the caller when the HPSS server or utility is invoked. This field is not enterable when the Authentication Type field is set to Keytab File.
 - **Keytab File.** The pathname of the keytab file to be created if the Authentication Type is set to "Keytab File". This file is normally located in the `/var/hpss/etc` directory. This field is not enterable when the Authentication Type field is set to Password.
6. Review and modify (if necessary) the following authorization fields:
- **Local Site Name.** The value is usually set to the full machine name of the local host which can be determined using the 'hostname' and 'domainname' commands.
 - **Local Realm Name.** The value is usually set to the "Local Site Name" all capitalized.
 - **Local Realm ID.** The field is set to a unique ID number for each site. Ask your support representative for an appropriate value.
 - **Realm URL.** This field is only needed for cross realm. Accept the default value.
 - **Administrator DN (Distinguished Name).** The administrator name that is allowed to add/update/remove entries in LDAP.
 - **Administrator Password.** The password used by the administrator to manage entries in LDAP.
 - **Verify Password.** Repeat of the LDAP administrator password entered to verify it was entered correctly.
 - **Enable Kerberos authentication.** This must be enabled. UNIX Authentication is not supported with LDAP Authorization
 - **Configure an LDAP server in this host.** The flag is set to create an LDAP instance locally on this host machine. If an LDAP server already exists, un-select this flag.
 - **Re-create DB2 Instance.** The flag is set to indicate that a new LDAP database is to be created. If an LDAP server and database already exist, un-select this flag.
 - **DB2 Instance Name.** The LDAP's DB2 instance owner.
 - **Database Name.** The name of the LDAP database. In most cases, the default value of 'ldaphpss' should be used.
 - **DB2 Instance Owner Password.** This is the UNIX password for the userid specified in the "DB2 Instance" field.
7. By default, the system's configuration files (`/etc/passwd`, `/etc/group`, and `/etc/shadow`) are used to administer the authentication and authorization services. As an option, the HPSS configuration files can be used instead. These files will be created by **mkhpss** as part of this configuration step. Other HPSS utilities are available to administer these HPSS configuration files. Refer to Section 2.2.2: *Security Mechanisms* in the *HPSS Management Guide* for more information. To use the HPSS configuration files, select the "Enable local

Unix Passwd Files" checkbox as shown in the example above. The default names for the files should be used as displayed.

8. Select the "Configure Server accounts" checkbox to create accounts for HPSS servers.
9. Click on the "Configure Security Services" button at the bottom of the screen to perform the specified security configuration.
10. If the configuration was a success, you should see the following message in the command output window:

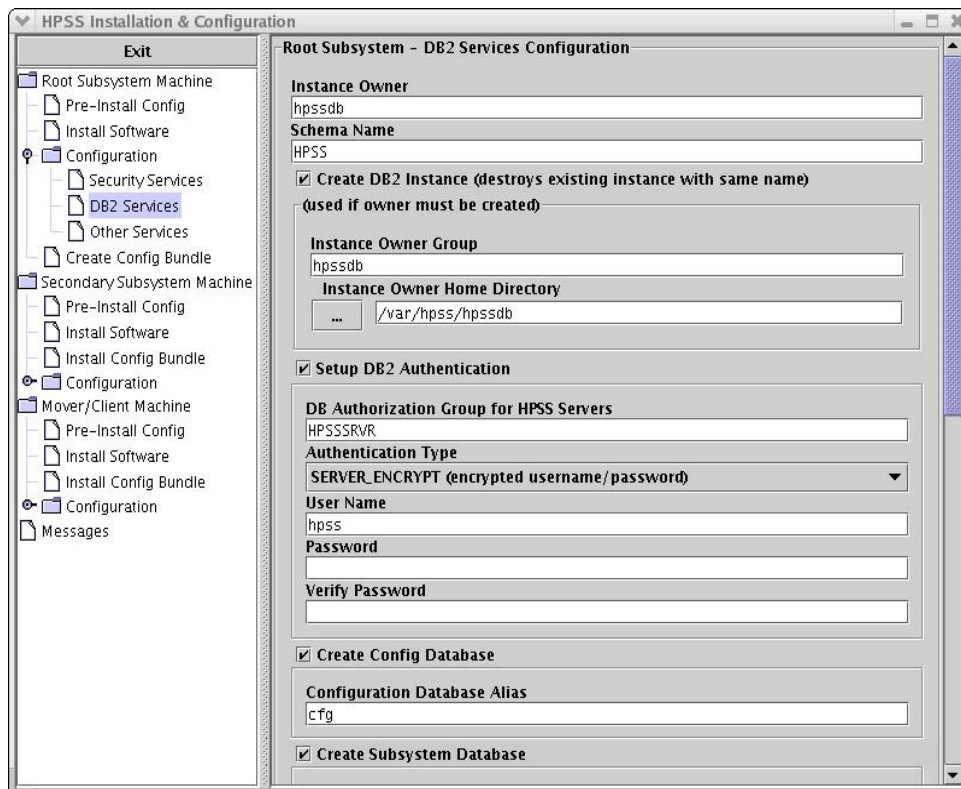
```
## run command exited with status 0
```

Click the 'Done' button on the Command Output window to close the window.

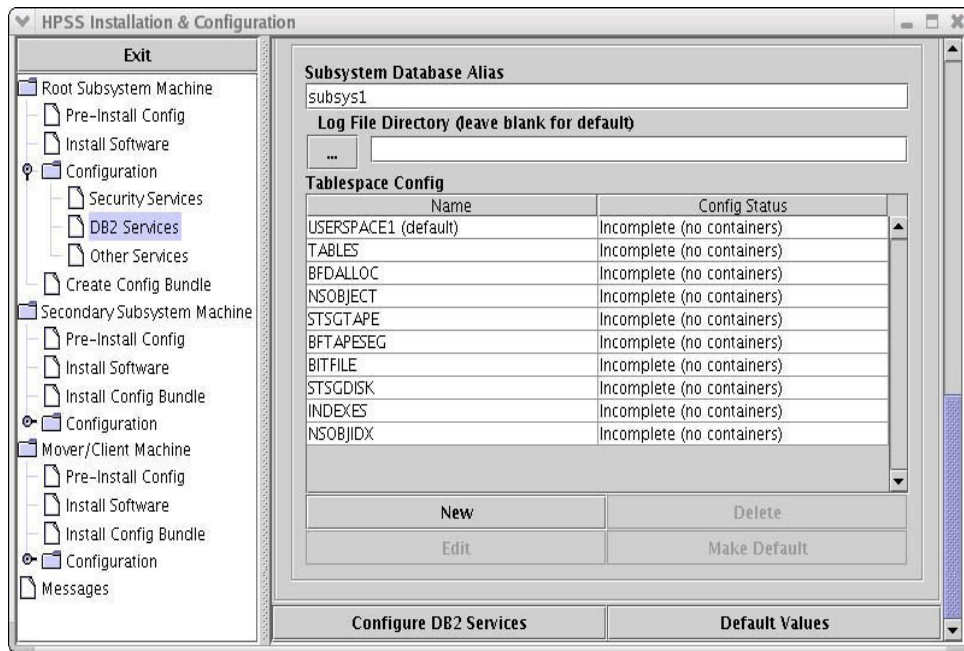
5.3.1.5. Configure DB2 Services

To configure DB2 databases to manage HPSS metadata, perform the following steps:

1. From the "Root Subsystem Machine" submenu in the left panel, click on the "Configuration" icon and then the "DB2 Services" menu item. The following window will be shown:



2. Move the scroll bar to review the remainder of the right hand panel, which looks like the following screen:



3. Review and modify (if necessary) the following fields:

- **Instance Owner.** The name of the DB2 instance owner, normally 'hpssdb'.
- **Schema Name.** The name of the DB2 schema containing the HPSS metadata table, normally 'hpss'.
- **Create DB2 Instance.** Select this checkbox to create the DB2 instance.
- **Instance Owner Group.** The UNIX group to which the Instance Owner is assigned, normally 'hpssdb'.
- **Instance Owner Home Directory.** The directory where the HPSS DB2 instance configuration files are stored, normally /var/hpss/hpssdb.
- **Setup DB2 Authentication.** Select this checkbox to setup DB2 authentication.
- **DB Authorization Group for HPSS Servers.** The UNIX group ID that is used to allow the HPSS servers and utilities to access the HPSS DB2 instance.
- **Authentication Type.** Currently there is only one authentication type allowed.
- **User Name.** The UNIX userid to be used for DB2 authentication.
- **Password and Verify Password.** Enter and verify the UNIX password for the userid entered in the User Name field.
- **Create Config Database.** Select this checkbox to create the "cfg" database. If this is the initial instance of HPSS and the configuration 'CFG' database has not been created, make sure the checkbox is selected.
 - ⚠ *If the 'CFG' database has already been created and you do not wish to destroy it, be sure to un-check this box. Otherwise, it will be destroyed and re-created and any configuration information will be lost.*
- **Configuration Database Alias.** The "real" database is configured as 'hcfg', while the database alias is the name used by HPSS servers and utilities to reference the database. Do not change the default value.

- **Create Subsys Database.** Select this checkbox to create the "subsys1" database.
- **Subsystem Database Alias.** The "real" database is configured as 'hsubsys1', while the database alias is the name used by HPSS servers and utilities to reference the database. The default value of 'subsys1' should normally be used.
- **Log File Directory.** The DB2 log files will be placed in the /var/hpss/hpssdb filesystem by default. The directory should be changed to /db2/log/cfg for the CFG database, and /db2/log/subsys1 for the SUBSYS1 database. To change the location of the log files, enter the full pathname in the text box.
- **Mirrored Log File Directory.** The directory should be changed to /db2/mirror-log/cfg for the CFG database, and /db2/mirror-log/subsys1 for the SUBSYS1 database. To change the location of the log files, enter the full pathname in the text box.
- **Tablespace Config.** The table shows the defined tablespace entries and their current configuration status. Each tablespace will contain one or more HPSS DB2 tables. By selecting a tablespace and using the 'edit' button, the tables assigned to the tablespace can be modified. The configuration status will change to 'OK' when the container assigned to the tablespace is valid.

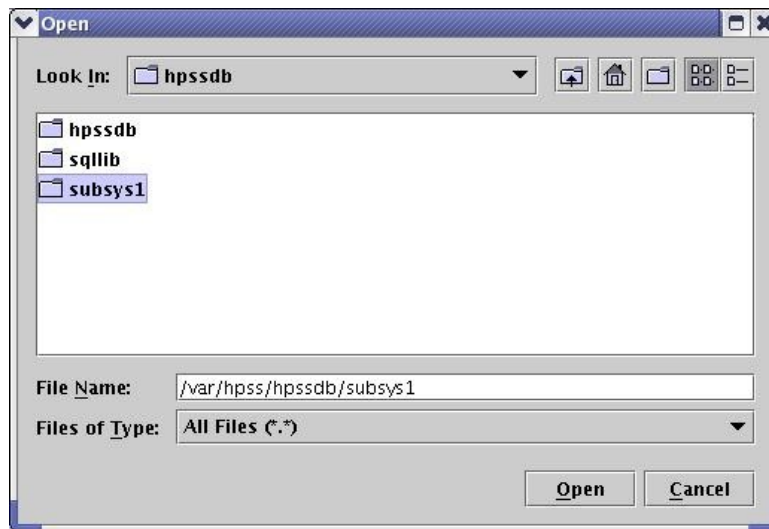
Perform the following steps for each tablespace:

- Under the subsection labeled "Tablespace Config", select a displayed tablespace and click the "Edit" button. This will open the "Tablespace Config" window for the selected tablespace as shown:

The screenshot shows a window titled "Tablespace Config" with the following fields and controls:

- Name:** USERSPACE1
- Page Size:** 8K
- Buffer Pool Pages:** 2048
- Tablespace Type:** System managed (filesystem)
- Container Paths - existing data will be destroyed!**: An empty list with "Add" and "Delete" buttons.
- Size of Each Device (e.g. 512M or 2G - leave blank for auto sizing):** An empty text field.
- Tables/Indexes:** A table with columns "Name" and "Type", and "Add" and "Delete" buttons.
- Buttons:** "OK" and "Cancel" at the bottom.

- B. On the “Tablespace Config” window, set the Tablespace Type from the drop down menu. For the USERSPACE1 tablespace, select “SMS” System Managed Space. For the remaining tablespace entries, select Tablespace Type to "DMS" Database Managed Space.
- C. For USERSPACE1, which uses System Managed space, from the "Tablespace Config" window, click the 'Add' button and select the directory you would like to allocate as the containers for this tablespace. This empty directory must be created before running this configuration option. The Add button will show the following screen:



- D. Select, or enter, the desired directory. Click the "OK" button to return you back the the main configuration window. The USERSPACE1 "Config Status" should now say "OK".
- E. For other tablespaces, using DMS (Database Managed Space), select the TABLES entry. The following screen will be shown:

Tablespace Config

Name
TABLES

Page Size
8K

Buffer Pool Pages
2048

Tablespace Type
Database managed (raw devices) - recommended

Container Paths - existing data will be destroyed!

Add Delete

Size of Each Device (e.g. 512M or 2G - leave blank for auto sizing)

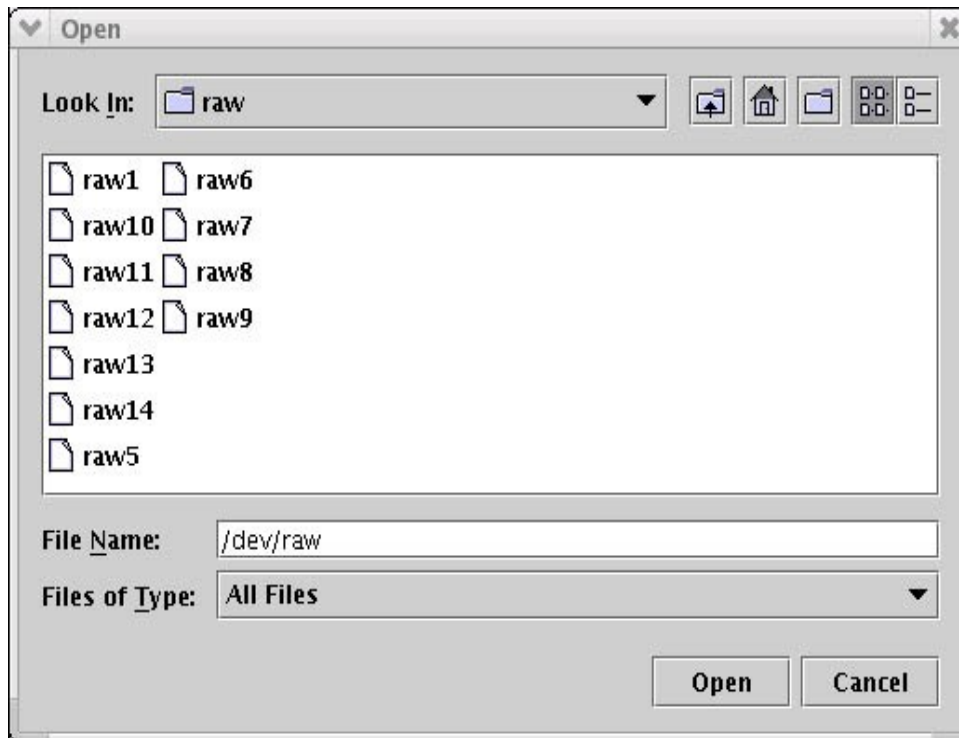
Tables/Indexes

Name	Type
ABSADDR	TABLE
ACCTLOG	TABLE
ACCTSNA	TABLE
ACCTSUM	TABLE
BFCOSCHANGE	TABLE
BFDISKSEG	TABLE
BFMIGRREC	TABLE
BFPURGEREC	TABLE
BFSSEGCHKPT	TABLE
BFSUNLINK	TABLE
DISKSPACE	TABLE
DMGFILESET	TABLE
MPSCHKPT	TABLE
NSACL	TABLE
NSFILESETATTR	TABLE
NSTEXT	TABLE
SSPVDISK	TABLE
SSPVTAPE	TABLE
STORAGEMAPDISK	TABLE

Add Delete

OK Cancel

- F. Select the "Add" button to add "raw devices" to Container Path list. The "raw devices" must already exist. For AIX, the logical volumes must have already been created and the "raw" form be specified (i.e. LV "dbs1.tables", specify /dev/rdbms1.tables") For Linux, not only must the partition exist, but the "raw" mapping must also be in place prior to selecting the container path. See Section 3.5.3: *HPSS Filesystems* on page 71 for a listing of these mappings. On Linux, the Container Path "Add" button will display the following screen:



4. When all the tablespaces show a status of "OK", click on the "Create DB2 Services" button at the bottom of the screen to perform the DB2 configuration.
5. If the configuration was a success, you should see the following message in the command output window:

```
## run command exited with status 0
```

Click the 'Done' button on the Command Output window to close the window.

5.3.1.5.1. Remote DB2 Client Access & Fileset Creation/Deletion

This information is pertinent to sites that have chosen to deny remote client access to DB2. The method for configuring DB2 in such a manner is outside the scope of this document, please refer to DB2 documentation and support for more information on such a configuration.

If, as part of configuring DB2 to deny remote access, a variable in the DB2 environment called DB2COMM has been unset, creation and deletion of filesets will fail inside of DB2. You must have

a variable named DB2_USE_LOCAL_RESYNC set to the value of 'true' when starting DB2 in order for the aforementioned fileset operations to complete successfully:

ssh & tcsh:

```
setenv DB2_USE_LOCAL_RESYNC true
```

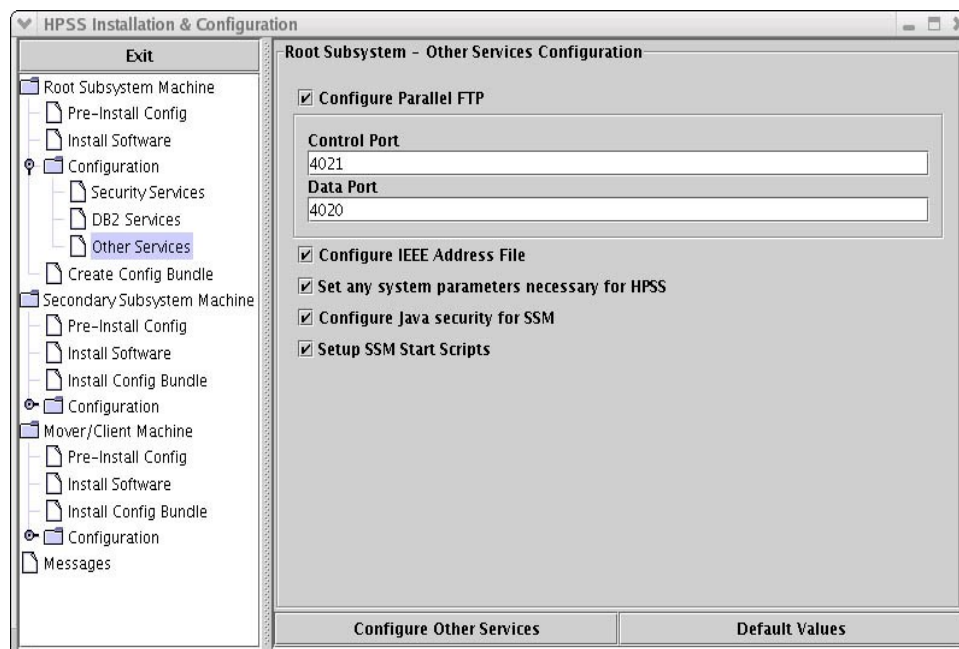
sh & bash:

```
export DB2_USE_LOCAL_RESYNC=true
```

5.3.1.6. Configure Other Services

This menu configures various services such as Parallel FTP, Java security for SSM, and SSM start scripts. To configure Other Services, perform the following steps:

1. From the "Root Subsystem Machine" submenu in the left panel, click on the "Configuration" icon and then the "Other Services" icon. The following window will be shown:



2. By default, all boxes are selected. If some items have already been configured, un-select the appropriate box to by-pass the re-configuration of that task. When satisfied with your selections, select the "Configure Other Services" button and verify the command output window returns with a status of "0". Depending upon the items selected, HPSS will be 1) setting up /etc/services and inetd to allow HPSS ftp/pftp daemon to be invoked on this system, 2) initialize the IEEE address file in /var/hpss/etc, 3) copy the SSM configuration template files to /var/hpss/ssm.
3. If the configuration succeeds, you will see the following message in the command output window:

```
## run command exited with status 0
```

Click the 'Done' **button** on the Command Output window to close the window.

4. After exiting **mkhps**, verify permissions on the generated files. In particular, note the permissions on the keytab files. The **hpss.keytab** is used by HPSS servers to establish credentials. The **mm.keytab** is used by hpss utility programs. The **kadm5.keytab** is used to establish credentials as the Kerberos admin. Be certain that the permissions on these files allow access only to the appropriate users.

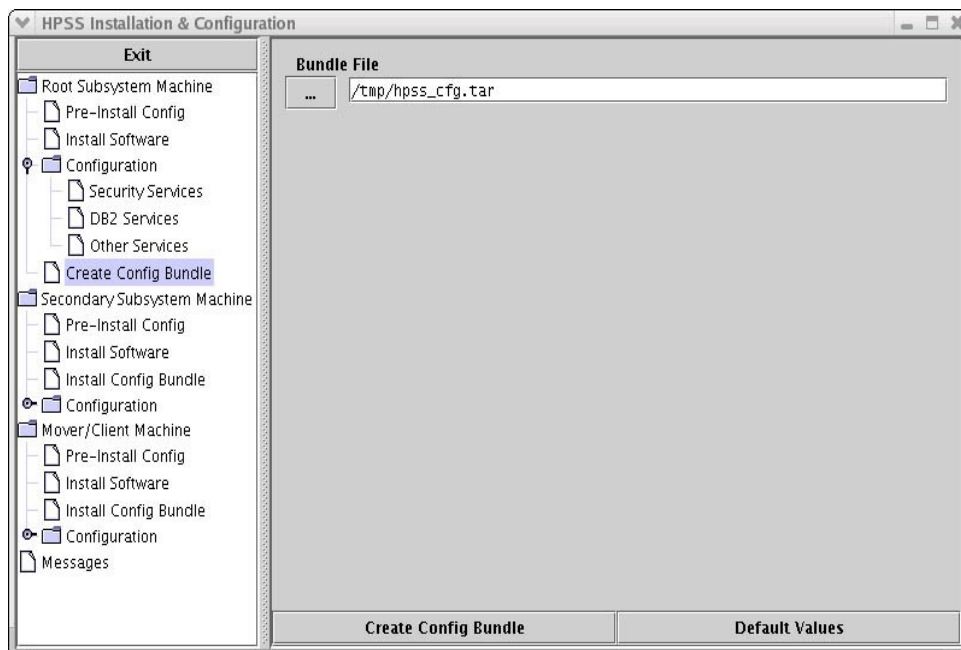
/var/hpss/etc/hpss.keytab
/var/hpss/etc/mm.keytab
/var/hpss/krb5kdc/kadm5.keytab

5.3.1.7. Create Configuration Bundle

To distribute the configuration information to other subsystem/mover nodes, use the “Create Config Bundle” option. This option should be used *only* after the Location Server has been defined/configured by the administrator to include the EndPoint information in the **ep.conf** file. Otherwise, the configuration bundle will not contain the endpoint information and the file will need to be copied/transferred manually to the other nodes after the Location Server is configured.

To create the HPSS configuration bundle perform the following steps:

1. From the "Root Subsystem Machine" submenu in the left panel, click on the "Create Config Bundle" icon. The following window will be shown:



2. Modify the "Bundle File" name, if desired.
3. Select "Create Config Bundle" button. If the configuration succeeds, you will see the following message in the command output window:

```
## run command exited with status 0
```

Click the 'Done' **button** on the Command Output window to close the window. Verify that the configuration bundle has been created.

5.3.2. Install and Configure HPSS - Secondary Subsystem Machine

For the secondary subsystem machine, the following configuration steps must be performed:

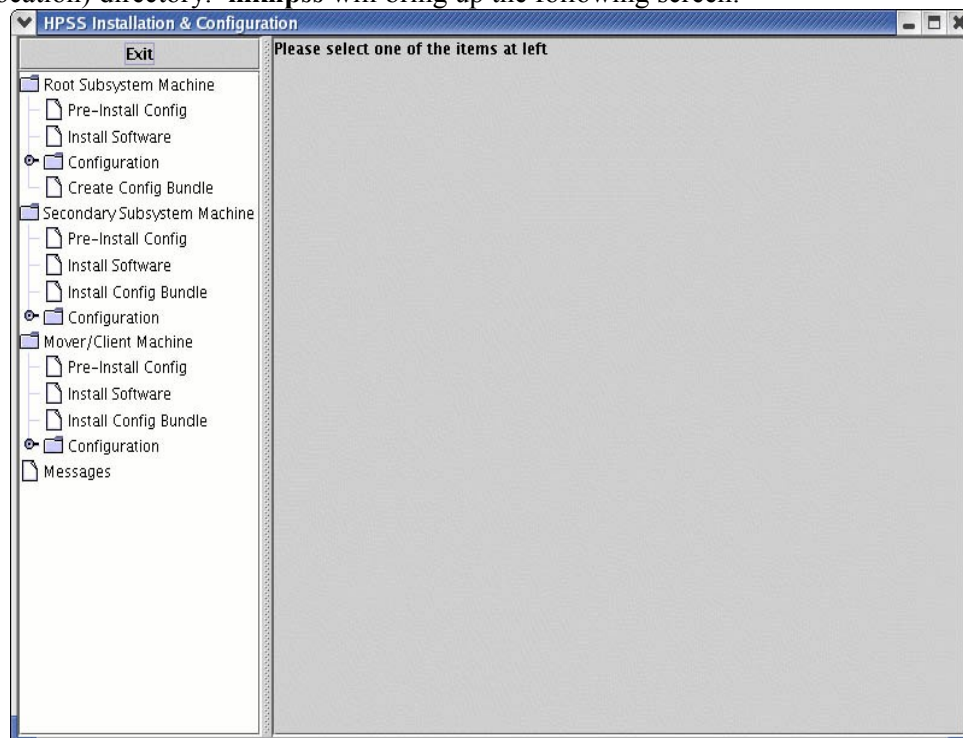
- Pre-installation Configuration
- Install HPSS documentation and DB2
- Set up DB2 permanent license
- Install configuration bundle
- Configure Security Services
- Configure DB2
- Configure Other Services

The following sections describe the procedure to perform the above steps.

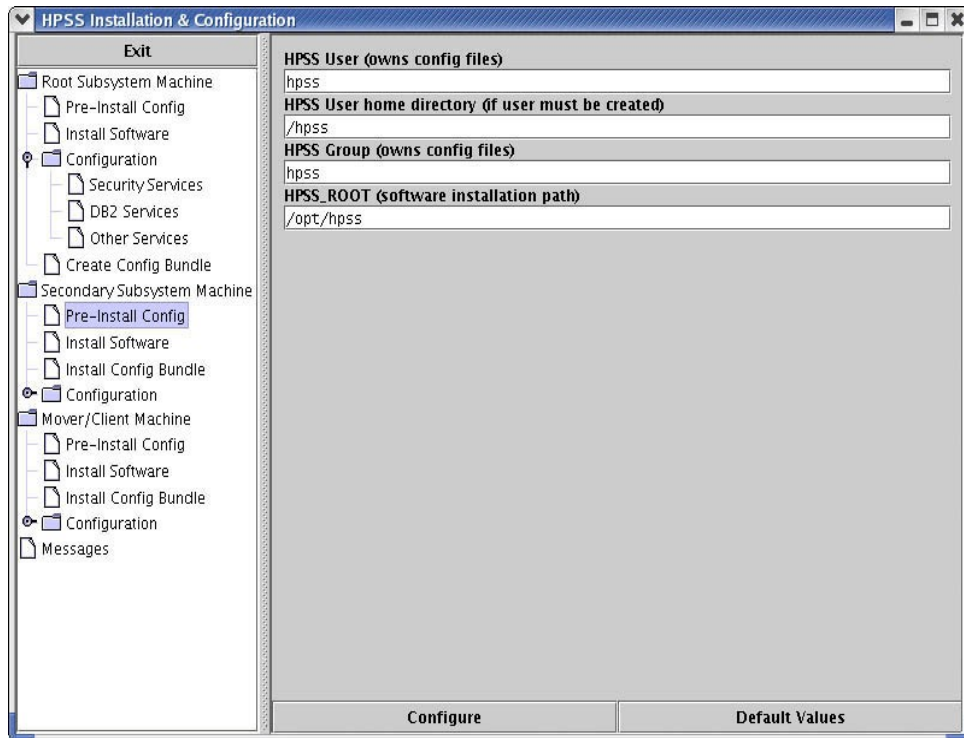
5.3.2.1. Pre-Installation Configuration

Before configuring HPSS, you will need to perform the following steps to setup the appropriate configuration directories and files on the machine where the secondary subsystem servers will run:

1. Install the HPSS software on the subsystem server node. For AIX, an LPP image or tar file will be provided in HPSS release 6.2.0. For Linux, an RPM package or tar file will be provided in HPSS release 6.2.0. Invoke the **mkhpss** utility from the /opt/hpss/bin (default location) directory. **mkhpss** will bring up the following screen:



2. From the "Secondary Subsystem Machine" submenu, click on the 'Pre-Install Config' icon in the left panel. **mkhpss** will display the following screen:



3. Verify that the default values are as desired. Modify them, if necessary. Click the 'Configure' button to perform the pre-installation setup.
4. If the pre-installation configuration was a success, you will see the following message in the command output window:

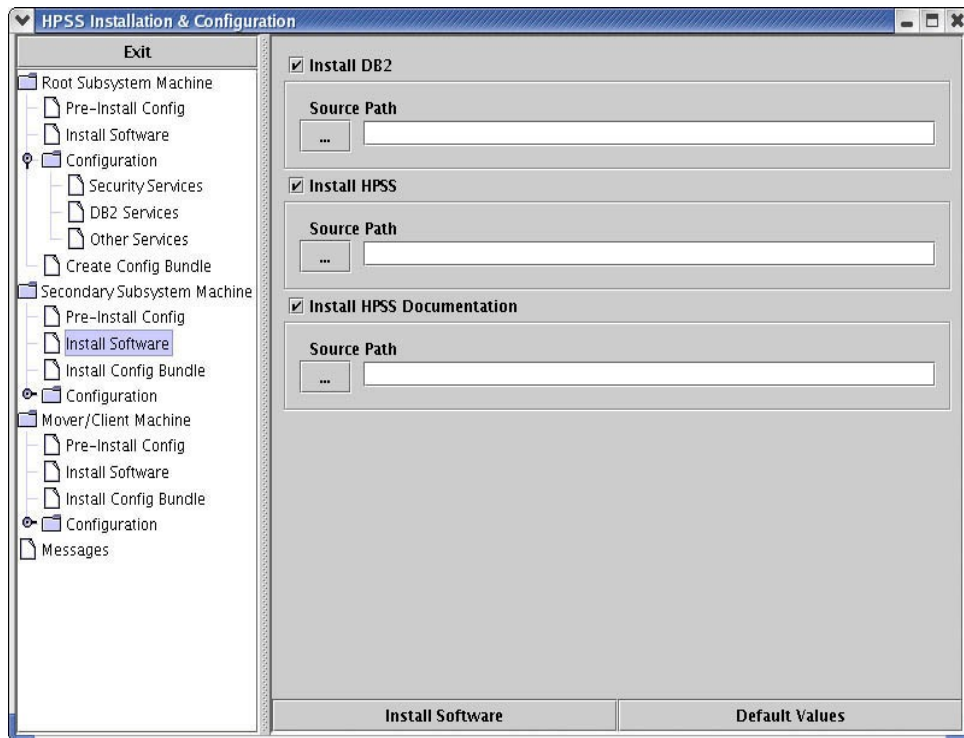
```
## run command exited with status 0
```

Click the 'Done' **button** on the Command Output window to close the window.

5.3.2.2. Install HPSS Documentation and DB2 Software on a subsystem

This section describes the procedure to install HPSS documentation and DB2 software on the secondary subsystem machine.

From the "Secondary Subsystem Machine" submenu in the left panel, click on the "Install Software" option. The right panel will be displayed as shown:



This panel allows you to install the DB2 software, HPSS software (option starting in 6.2.1), and HPSS documentation. Perform the following steps to install the software:

1. To install DB2, check the 'Install DB2' checkbox. Click on the 'Source Path' button and select the directory that contains the DB2 filesets you wish to install. DB2 software will be installed in the /usr/opt/db2_08_01 (AIX) or /opt/IBM/db2/v8.1 (Linux) directory.
2. For 6.2.0, the HPSS software and documentation should already be installed according to the installation step in section 5.3.2.1. Installing from CD will be an option starting in 6.2.1.
3. For 6.2.0, the **hpssuser** utility should be used to package the HTML files for delivery to each **hpssgui** machine or to a common shared file system on a remote node. Installing from CD will be an option starting in 6.2.1.



To provide the SSM Help facility, the HTML files from the HPSS documentation must be accessible from the node where the hpssgui program runs. The files may be installed on a shared file system available to all nodes, or the hpssuser utility may be used to package the HTML files for delivery to each hpssgui machine.

4. Click the 'Install Software' button to install the selected software.
5. If the installation was a success, you should see the following message in the command output window:

```
## run command exited with status 0
```

Click the 'Done' **button** on the Command Output window to close the window.

5.3.2.3. Set Up DB2 Permanent License

This section describes the procedure to create the DB2 permanent license and to specify the number of processors licensed for the secondary subsystem machine.

To create a permanent DB2 license, issue the following commands:

```
% su -
% cd /usr/opt/db2_08_01/adm
% ./db2licm -a <path name to the DB2 generic license file>
```

The generic DB2 license file (*/*db2/license/db2ese.lic) can be found on the DB2 Installation CD or image. It can also be obtained by contacting your HPSS Support Representative.

To update the license with the appropriate number of of processors, issue the following command:

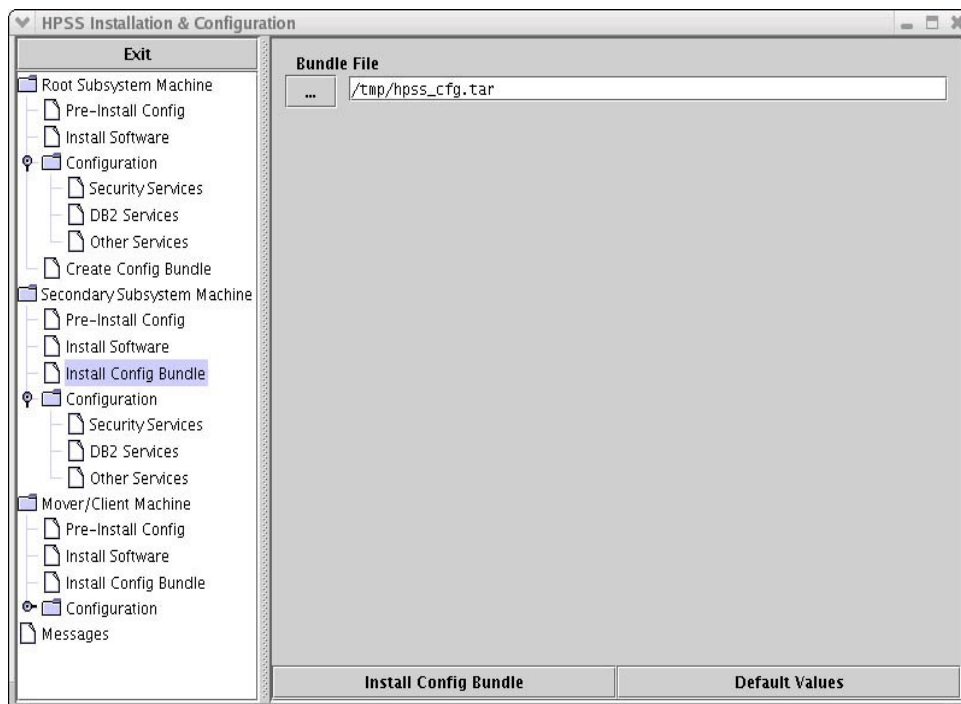
```
% ./db2licm -n db2ese <number of processors>
```

Refer to the DB2 Command Reference document for more information on how to use the db2licm utility to manage the DB2 license.

5.3.2.4. Install Configuration Bundle

To install the configuration bundle, perform the following steps:

1. Copy the configuration bundle from the root subsystem machine and place it into a directory with adequate space, such as /tmp.
2. From the "Secondary Subsystem Machine" submenu, select the "Install Config Bundle" option. The following screen will be shown:



3. Verify that the "Bundle File" field has the correct filename for the configuration bundle file.
4. Click on the "Install Config Bundle" button to begin the installation process. If the installation was a success, you should see the following message in the command output window:

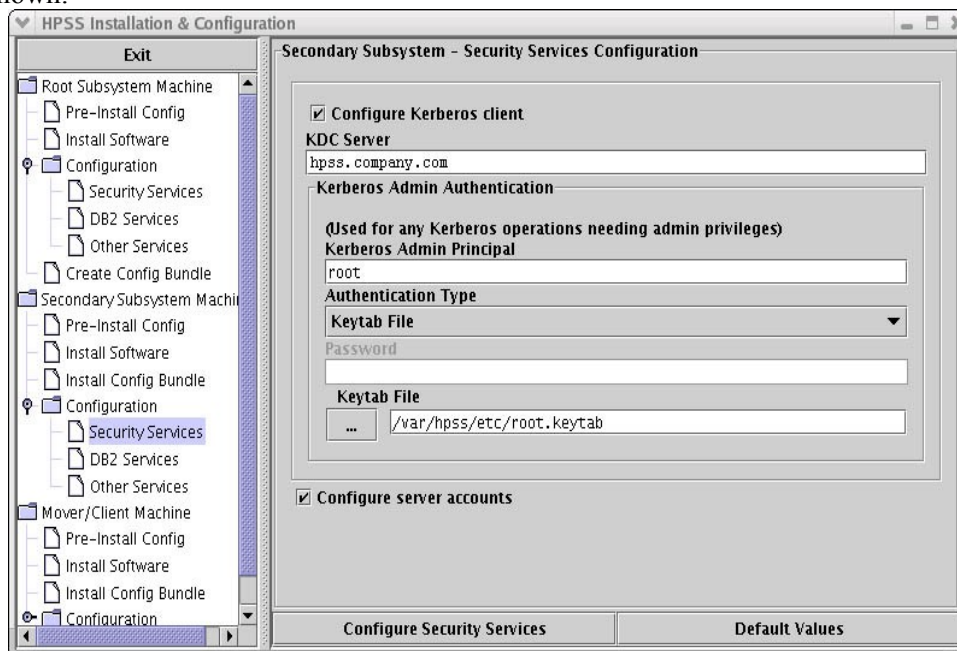
```
## run command exited with status 0
```

5.3.2.5. Configure HPSS Security Services

This section describes the procedure to configure a Kerberos or UNIX Security Client on the secondary subsystem machine. The security client must match the security mechanism on the root machine.

To configure the security client for the secondary subsystem machine, perform the following steps:

1. From the Secondary Subsystem Machine submenu in the left panel, click on the "Configuration" icon and then the "Security Services" icon. The following screen will be shown:




2. To configure the Kerberos client, select the "Configure Kerberos Client" checkbox. Review and update the following fields if necessary:
 - **KDC Server.** This is the fully qualified name of the host that is running the Kerberos server.
 - **Kerberos Admin Principal.** The name of the administration principal that is used to manage the Kerberos server.
 - **Authentication Type.** There are two supported options: Keytab File or Password. The Keytab File option allows HPSS servers or utilities to read a keytab file to authenticate. The Password option requires a password to be supplied each time an HPSS server or utility is invoked.
 - **Password.** The password used to authenticate the caller when the HPSS server or utility is invoked. This field is not enterable when the Authentication Type is Keytab File.
 - **Keytab File.** The pathname of the keytab file to be created if the Authentication Type is set to "Keytab File". This file is usually stored in the /var/hpss/etc directory. This field is not enterable when the Authentication Type is Password.

- **Configure server accounts.** This checkbox is flagged when the server accounts should be created on the local machine. This is usually not required when the Kerberos server has already been configured with the HPSS server accounts.
3. To configure the UNIX client, un-check the "Configure Kerberos Client" checkbox and select the "Configure server accounts" checkbox.
 4. Click on the "Configure Security Services" button to configure the security client on the Secondary Subsystem machine.
 5. If the configuration was a success, you should see the following message in the command output window:

```
## run command exited with status 0
```

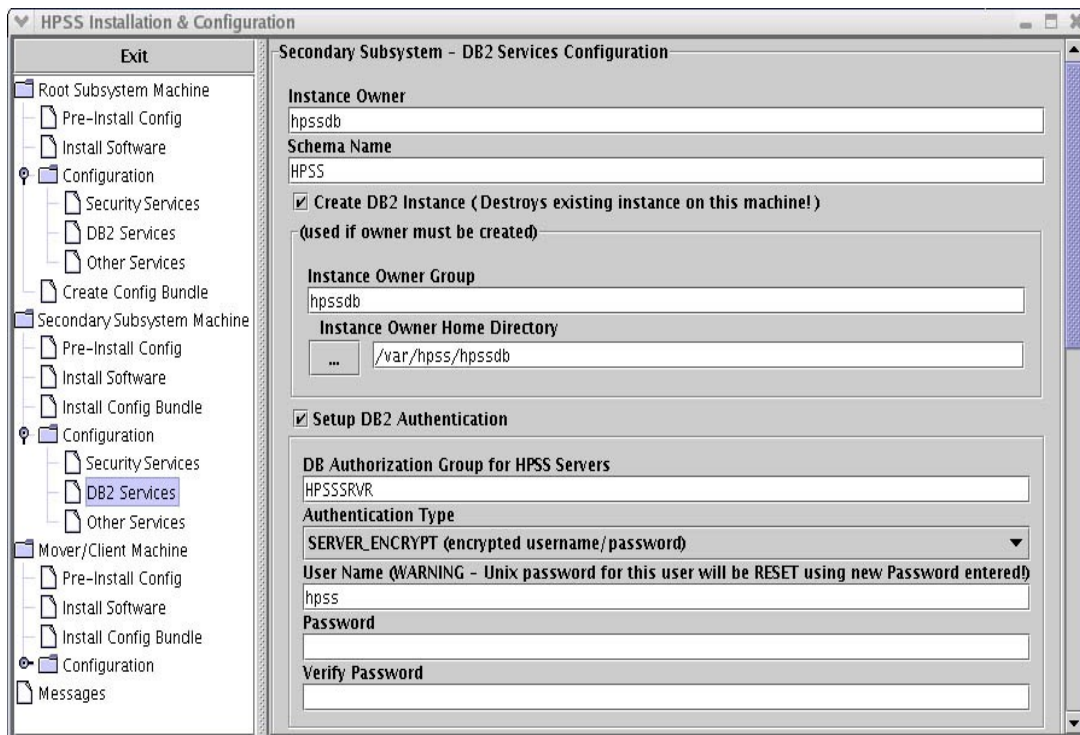
Click the 'Done' **button** on the Command Output window to close the window.

5.3.2.6. Configure DB2 Services

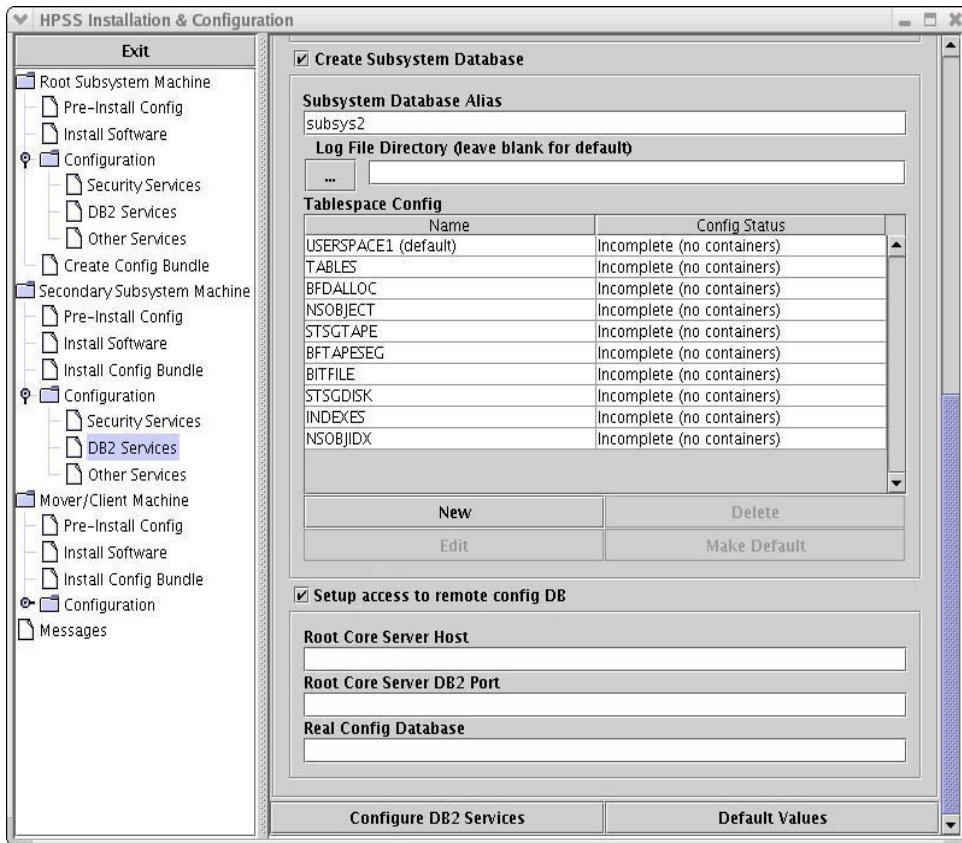
 *Ensure that you are running **mkhps** on the secondary subsystem machine. Executing these steps on the root subsystem will destroy existing data!*

To configure the DB2 subsysX database to manage HPSS metadata for the secondary subsystem, perform the following steps:

1. From the "Secondary Subsystem Machine" submenu in the left panel, click on the "Configuration" icon and then the "DB2 Services" icon. The following window will be shown:



- Using the scrollbar, move the right-hand panel until you see the following screen:



- Review and modify (if necessary) the following fields:

- **Instance Owner.** The name of the instance owner HPSS will use. Should use the default value of 'hpssdb' in most cases.
- **Schema Name.** The name of the HPSS schema used by the database to reference the HPSS DB2 tables. Should use the default value of 'HPSS' in most cases.
- **Create DB2 Instance.** Select this checkbox to create the DB2 instance on this machine. This option has no effect on the instance of another machine, only for this local node.
- **Instance Owner Group.** The UNIX group assigned to the HPSS database files. Should use the default of 'hpssdb' in most cases.
- **Instance Owner Home Directory.** The directory where the HPSS DB2 instance configuration files are stored. In most cases, the default directory of /var/hpss/hpssdb should be used.
- **Setup DB2 Authentication.** Select this checkbox to setup DB2 authentication locally on this machine. This option does not effect the authentication setup on any other machine.
- **DB Authorization Group for HPSS Servers.** The UNIX group that is used to allow the HPSS servers and utilities to access the HPSS DB2 instance.

- **Authentication Type.** Currently there is only one authentication type allowed.
 - **User Name.** The UNIX userid to be used for DB2 authentication. This userid must have a valid UNIX account before running this configuration option.
 - **Password and Verify Password.** Enter and verify the UNIX password for the userid entered in the User Name field above.
 - **Create Subsys Database.** Select this checkbox to create the "subsysX" database (where X will be 2, 3, ...).
 - **Subsystem Database Alias.** The "real" database is configured as 'hsubsysX' while the database alias is the name used by HPSS servers and utilities to reference the database. The default value of 'subsysX' should normally be used (for example, subsys2 for subsystem #2), but will need to be incremented for additional subsystems.
 - **Log File Directory.** The DB2 log files will be placed in the /var/hpss/hpssdb filesystem by default. The directory should be changed to /db2/log/cfg for the CFG database, and /db2/log/subsys1 for the SUBSYS1 database. To change the location of the log files, enter the full pathname in the text box.
 - **Mirrored Log File Directory.** The directory should be changed to /db2/mirror-log/cfg for the CFG database, and /db2/mirror-log/subsys1 for the SUBSYS1 database. To change the location of the log files, enter the full pathname in the text box.
 - **Tablespace Config.** Define the DB2 tablespace and containers for the subsysX database. See the Tablespace Config options discussed in the Root Subsystem Machine section for details related to adding/modifying tablespace and containers on the Secondary Subsystem Machine.
 - **Setup access to remote config DB.** This checkbox should remain selected (default) to allow connectivity to the root subsystem database.
 - **Root Core Server Host.** The hostname of the machine running the HPSS DB2 instance (if not local)
 - **Root Core Server DB2 Port.** The port number the local host should contact the remote host by to access the DB2 database.
 - **Real Config Database.** The name of the database to be accessed by the local host. The name will be 'hXXXX' rather than the non-h name.
4. When all data fields have been updated, click on the "Create DB2 Services" button at the bottom of the screen to perform the DB2 configuration.
 5. If the configuration was a success, you should see the following message in the command output window:

```
## run command exited with status 0
```

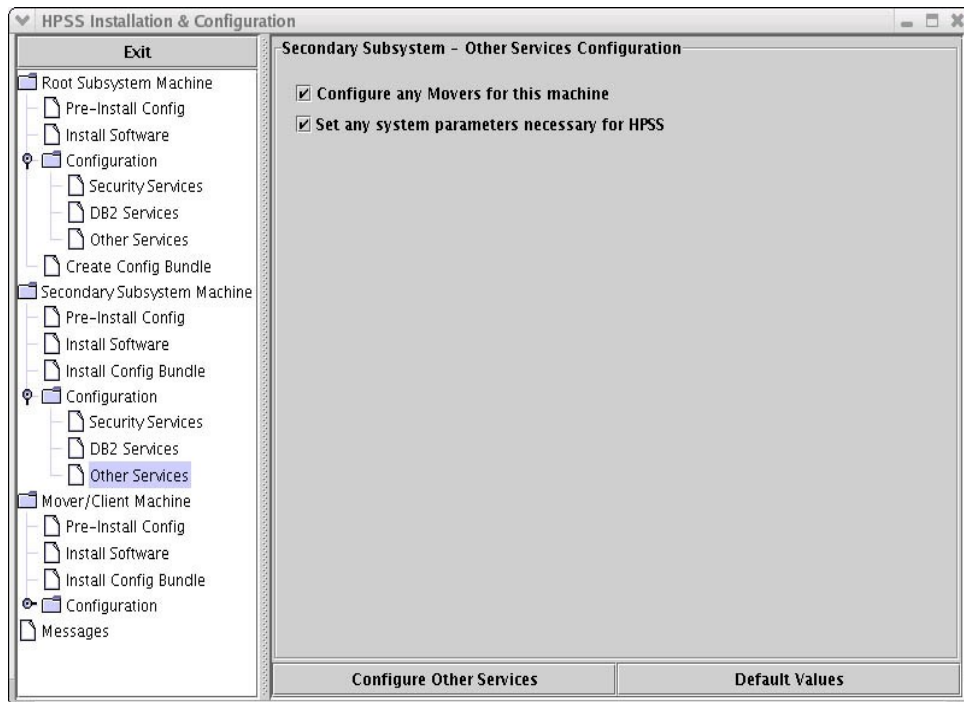
Click the 'Done' button on the Command Output window to close the window.

5.3.2.7. Configure Other Services

To configure Other Services on the secondary subsystem machine, perform the following steps:

1. From the "Secondary Subsystem Machine" submenu in the left panel, click on the

"Configuration" icon and then the "Other Services" icon. The following window will be shown:



2. Click on the "Configure Other Services" button to complete the configuration of the Secondary Subsystem Machine. If the configuration was a success, you should see the following message in the command output window:

```
## run command exited with status 0
```

Click the 'Done' **button** on the Command Output window to close the window.

5.3.3. Install and Configure HPSS - Mover/Client Machine

For the Mover/Client machine, the following configuration steps must be performed:

- Install Mover/Client source code
- Install configuration bundle
- Create /var/hpss subdirectories
- Modify Kerberos configuration, if necessary
- Check time and IP address

The following sections describe the procedure to perform the above steps.

5.3.3.1. Install Mover/Client source code

Follow the instruction in on page 174, section 5.8.2 Construct and Build the HPSS Mover/Client

Source Tree to obtain the source code from the root machine and install on the Mover/Client machine.

5.3.3.2. Install Configuration Bundle

The configuration bundle is created on the root subsystem and contains configuration files used by the root, Mover/Client and subsystem machines. See page 159, section 5.3.1.7 Create Configuration Bundle for instructions on creating the Configuration Bundle on the root machine.

To install the configuration bundle, perform the following steps:

1. Copy the configuration bundle from the root subsystem machine and place it into a directory with adequate space, such as /tmp.
2. Untar the file which will populate the /var/hpss/etc directory tree of the Mover/Client machine.
3. Verify the paths to the library files in /var/hpss/etc/auth.conf and /var/hpss/etc/authz.conf are using the correct path to the HPSS tree.

5.3.3.3. Create /var/hpss subdirectories

Change directory to the /var/hpss directory. Using mkdir, create the /var/hpss/cred and /var/hpss/tmp directories.

5.3.3.4. Modify Kerberos Configuration File, If Necessary

Either copy the /etc/krb5.conf file from the root machine or modify the Mover/Client's /etc/krb5.conf file so that it is using the root machine is listed as the default_realm and is listed in the realms and domain_realm sections.

5.3.3.5. Check Time and IP Address

Compare the time on the root machine with the time on the Mover/Client machine using the date command. The time difference must be less than 5 minutes for the authentication between the machines to succeed.

Additionally, if the Operating System was newly installed on the Mover/Client machine, verify that the /etc/hosts file contains the IP address, not just the loopback address.

5.4. Post Installation Procedures

After the HPSS software has been installed and its infrastructure has been configured, perform the following verifications:

1. Assuming default installation and configuration, verify the following directories have been created:

/opt/hpss/bin/<HPSS binaries>

/opt/hpss/lib/<HPSS libraries>

/opt/hpss/include/<HPSS include and idl files>

/opt/hpss/msg/<HPSS message catalog>

/opt/hpss/tools/<HPSS tools>

/opt/hpss/man/<HPSS man pages>

/opt/hpss/config/<HPSS configuration scripts>
/opt/hpss/stk/<STK files>
/opt/hpss/src/<HPSS source files> *Included only if the hpss-src package is installed.*
/var/hpss/<HPSS configuration files>

2. Verify that the HPSS file ownerships and file permissions are set as follows:

- Executable files: rwxr-xr-x bin bin
- Include files: r--r--r-- bin bin
- Library files: r--r--r-- bin bin
- Source files: r--r---- hpss hpss
- Make files: rw-rw-r-- bin bin
- Configuration files rwxrwxr-x hpss hpss



In particular, note that permissions on /var/hpss/etc/mm.keytab control the access to HPSS from many utility programs. Any user who can read mm.keytab will have permission to read and write directly into the HPSS database.

3. Verify that the DB2 permanent license has been set up by issuing the following commands:

```
% su -  
% /usr/opt/db2_08_01/adm/db2licm -l
```

Refer to Section 5.3.1.3: Set Up DB2 Permanent License on page 142 for more information on how to set up the DB2 license for an HPSS system.

5.5. HPSS Documentation & Manual Page Setup

This section describes the HPSS documentation provided to aid in the administration of the HPSS system as well as providing the basis for SSM help facility. It also describes the procedure to set up manual pages for HPSS utilities.

5.5.1. Documentation and SSM Help Package

The HPSS documentation is available via the HPSS website as a tar file and in PDF format. The HPSS documentation includes the following:

- HPSS Installation Guide
- HPSS Management Guide
- HPSS Error Messages Manual
- HPSS Programmer's Reference
- HPSS User's Guide

The HTML version of the Management Guides also serves as the source for the SSM Help feature. These HTML files must be accessible from each host from which the SSM hpssgui program is executed in order to be available in the SSM Help feature. The hpssuser program can be used to

bundle the HTML files for delivery to the hpssgui host machine. The recommended installation location for the HTML files on each hpssgui host is /var/hpss/doc for AIX and Linux platforms and c:\hpss\doc for Windows platforms.

5.5.2. Manual Page Setup

Perform the following steps to set up HPSS manual pages:

1. Create a symbolic link for the HPSS manual catalog. Assuming that the HPSS tree is installed at /opt/hpss, use the following commands:

```
% su - root
% cd /opt/hpss
% ln -s man cat7
```

2. Edit the /etc/environment file so that users will have the HPSS manual pages in their MANPATH. Assuming that the HPSS tree is installed at /opt/hpss, add the following line to the end of the /etc/environment file:

```
MANPATH=${MANPATH} : /opt/hpss
```

After this is done, users who subsequently login to the system will be able to view HPSS manual pages using the man command. For example, to view the **lshpss** manual page:

```
% man lshpss
```

5.6. Define HPSS Environment Variables

While most, if not all HPSS environment variables can be used as defined by HPSS, they should be reviewed to ensure that they are set to reflect your environment. The HPSS environment variables and their default values are defined in /opt/hpss/include/hpss_env_defs.h file. See Appendix E: *hpss_env_defs.h* which lists the default values of all HPSS environment variables. These environment variables may be overridden in /var/hpss/etc/env.conf or in the system /etc/environment file

The **/opt/hpss/config/hpss_set_env utility** can be used to insert environment variables into the /var/hpss/etc/env.conf file, list the values of the current HPSS environment variables, or list all HPSS default environment values:

Usage:

```
hpss_set_env [-all] [-def] [-set ENVNAME=NEWVALUE] |
              ENVNAME [ENVNAME...]
```

Where:

```
-all           Show current HPSS environment values
-def          Show all HPSS default environment values
-set          Set HPSS default environment values
ENVNAME       Display current value for ENVNAME
```

5.7. Tune DB2

Database tuning is an important aspect of maximizing HPSS performance, but it is a complex topic, requiring study and experimentation, to do well. **mkhps** creates initial configuration parameters for HPSS DB2 instances that we find to be effective for most systems, but additional tuning to deal with the specifics of a given HPSS installation may improve database performance. Administrators

wishing to learn more about DB2 tuning are referred to the *HPSS DB2 Tuning Guide*, available from your support representative, the *DB2 Administrative Guide: Performance*, available on-line from the IBM DB2 website, the *IBM DB2 Tuning Redbook*, and the many IBM and after-market books available on this subject.

Be sure to contact your HPSS support representative for advice before making changes to DB2 configuration parameters. See also Chapter 14: *Backup and Recovery* of the *HPSS Management Guide* for additional information.

5.8. Install and Build HPSS Source Code



The HPSS distribution package includes all binaries required by HPSS. However, due to special circumstances, it may be necessary for a site to rebuild the binaries. Contact your HPSS Support Representative to discuss your requirement for the HPSS source code.

This section is only applicable if you must work with the HPSS source code.

If rebuilding the HPSS binaries is required, the following should be performed:

- Construct and build the HPSS base source tree
- Construct and build the HPSS Mover/Client source tree
- Construct and build the HPSS HDM source tree (if XFS will be used)

5.8.1. Construct and Build the HPSS Base Source Tree

The HPSS base source tree contains the source code for all the HPSS components except the STK PVR proprietary code. If you plan to use the STK PVR, contact your HPSS Support Representative for information on how to obtain the STK PVR proprietary code.

5.8.1.1. Construct the HPSS Source Tree

To construct the HPSS base source tree, perform the following steps:

1. Log on as root.
2. Install the HPSS source code package (code is installed in the `/opt/hpss` directory).

5.8.1.2. Build the HPSS Base Source Tree

HPSS makes use of shared libraries. This allows the shared libraries to be compiled without requiring the executables to be relinked. The pathname to the shared libraries with which each server links is stored in its executable file. The executable will fail to load if the pathname of the shared library has been changed since it was linked. The location for all HPSS shared libraries is `/opt/hpss/lib`.

The `RUNLIBS_PATH` macro specifies the run-time top level directory of the HPSS tree. It defaults to `"/opt/hpss"`. HPSS defines its shared library locations in terms of this macro, as `$(RUNLIBS_PATH)/lib`.

To compile the HPSS sources, the following steps must be performed:

1. Change directory to `/opt/hpss` directory.
2. Review the `/opt/hpss/Makefile.macros` file. This file defines the "make" environments and options for the HPSS software compilation. Ensure that the environments and options are specified properly before running the "make" command.
3. Some important environment and option variables (with default values) are listed here:

```

Makefile.macros:
BUILD_PLATFORM = AIX
BUILD_TOP_ROOT = /opt/hpss
KRB5_AUTH_SUPPORT=on
UNIX_AUTH_SUPPORT=on
LDAP_AUTH_SUPPORT=off
GSI_AUTH_SUPPORT=off
CONVERSION_FROM_45 = off
CONVERSION_FROM_51 = off
BUILD_UNSUPPORTED = off
MVR1_PROGRAM_NAME = $(LOCAL_BIN)/hpss_mvr_tcp
MVR1_OPTIONS      = ""
# MVR2_PROGRAM_NAME = $(LOCAL_BIN)/hpss_mvr_ssd
# MVR2_OPTIONS      = "SSD"
# MVR3_PROGRAM_NAME = $(LOCAL_BIN)/hpss_mvr_omi
# MVR3_OPTIONS      = "OMI"
# MVR4_PROGRAM_NAME = $(LOCAL_BIN)/hpss_mvr_dd2
# MVR4_OPTIONS      = "DD2"

```

```

Makefile.macros.AIX:
PVR_LIST          = operator
AIX_JAVA_ROOT     = /usr/java14
DB_INSTALL_PATH  = /usr/opt/db2_08_01
LDAP_INSTALL_PATH = /usr/ldap
KRB5PATH         = /usr/local
GSIPATH          = /usr/gt2

```

```

Makefile.macros.LINUX:
PVR_LIST          = operator
LINUX_JAVA_ROOT   = /usr/java/j2sdk1.4.2_05
DB_INSTALL_PATH  = /opt/IBM/db2/v8.1
LDAP_INSTALL_PATH = /usr/ldap
KRB5PATH         = /usr/local
GSIPATH          = /usr/gt2

```

4. Log on as hpss.
5. Issue the "make clobber" command.
6. Issue the "make" command.

5.8.1.3. Generate and Bind the DB2 Helper Program

When filesets are created or updated, it is sometimes necessary to make entries in both the global and the subsystem database. When updating both of these databases, it is very important that the update be performed atomically. So, to accomplish an atomic update, a DB2 helper program is created. This DB2 helper program is 'run' by DB2 whenever it needs to perform an atomic update of the global and subsystem databases when creating or updating a fileset.

To generate this DB2 helper program and bind it to DB2 a script named **hpss_db2_bindall.ksh** is provided.

Whenever the base source tree is rebuilt, perform the follow steps to generate and bind the DB2 helper program:

1. Log on as hpss.
2. Change directory to /opt/hpss/bin.
3. Run the following command:

```
% hpss_db2_bindall.ksh
```

5.8.2. Construct and Build the HPSS Mover/Client Source Tree

This section describes the procedures to extract the Mover/Client code from the HPSS base source tree and to build the Mover/Client binaries.

5.8.2.1. Construct the HPSS Mover/Client Source Tree

The HPSS Mover/Client source tree can be extracted from the HPSS base source tree. The source tree should be completely compiled before attempting this procedure. To construct the Mover/Client component source tree, the following steps must be performed:

On the machine where the HPSS base source tree is installed:

1. Log on as hpss.
2. Change directory to the HPSS base source tree (the default location is /opt/hpss).
3. Review the Makefile.macros file.
4. Ensure that the target directory tree (where the source tree will be constructed) is completely empty.
5. Issue the following command:

```
% make BUILD_ROOT=<target directory tree> build-mvr [build-clnt] [build-ftp] [build-fs]
```

6. tar up the target directory tree

On the machine where the HPSS Mover/Client source tree will be constructed:

1. Log on as hpss.
2. Copy the tar file for the Mover/Client source to the Mover/Client machine.
3. Change directory to the /opt/hpss directory.
4. Untar the tar file for the Mover/Client source into the /opt/hpss directory.

5.8.2.2. Build the HPSS Mover/Client Source Tree

To compile the HPSS Mover/Client sources, the following steps must be performed:

1. Change directory to /opt/hpss directory.
2. Review the /opt/hpss/Makefile.macros file. This file defines the "make" environments and options for the HPSS software compilation. Ensure that the environments and options are specified properly.
3. Run the "make mvr [clnt][ftp]" command. See Section 13.4.3: *Installing the HPSS VFS Interface* of the *HPSS Management Guide* for direction to build and install the HPSS VFS Interface.

5.8.3. Construct and Build the HPSS HDM Source Tree



XFS is not supported in HPSS 6.2. XFS references have been left in the HPSS documentation to support the option of re-enabling XFS in future releases.

This section describes the procedures to extract the HDM source code from the HPSS base source tree and to build the HDM binaries.

5.8.3.1. Construct the HPSS HDM Source Tree

The HPSS HDM source tree can be extracted from the HPSS base source tree. To construct the HDM component source tree, the following steps must be performed:

On the machine where the HPSS base source tree is installed:

1. Log on as root.
2. Change directory to the HPSS base source tree (the default location is /opt/hpss).
3. Review the Makefile.macros file.
4. Ensure that the target directory tree (where the source tree will be constructed) is empty.
5. Issue the following command:

```
% make BUILD_ROOT=<HDM source tree directory> build-hdm
```

On the machine where the HPSS HDM source tree will be constructed:

1. Log on as root.
2. Copy the tar file for the Mover/Client source to the Mover/Client machine.
3. Change directory to the /opt/hpss directory.
4. Untar the tar file for the Mover/Client source into the /opt/hpss directory.

5.8.3.2. Build the HPSS HDM Source Tree

To compile the HPSS HDM sources, the following steps must be performed:

1. Change directory to /opt/hpss directory.
2. Review the /opt/hpss/Makefile.macros file. This file defines the "make" environments and options for the HPSS software compilation. Ensure that the environments and options are specified properly before running the "make hdm" command.

5.9. Supporting Both Unix and Kerberos Authentication for SSM

Once security services have been configured for your system (see Section 5.3.1.4: *Configure HPSS Security Services* on page 143 for details), if both Unix and Kerberos have been set up, it is possible to configure your system to support both Unix and Kerberos authentication for the SSM. If you can login to the system using the SSM gui, you can use it to set this up, as described in Section 3.3.1: *Configuring the System Manager Authentication for SSM Clients* of the *HPSS Management Guide*. If that is not an option because no available authentication mechanism is configured, you can use the procedure that follows to set up support for both authentication mechanisms. The combination of

Unix authentication with LDAP authorization is not supported at this time, so it only makes sense to do this if you are using Unix authorization.

To set up support for both authentication mechanisms, change the following fields in DB2:

<i>Table</i>	<i>Field</i>	<i>Old</i>	<i>New</i>	<i>Where</i>
server	NUM_AUTH_MECHS	1	2	desc_name = 'SSM System Manager'
server	AUTHN_MECHS1_MECHANISM	0	2	
server	AUTHN_MECHS1_AUTH_TYPE_KEY	0	1	
serverinterface s	AUTHN_MECH_SET_NUM_MECHS	1	2	server_id = (select server_id from server where desc_name = 'SSM System Manager') and descriptive_name = 'Administrative Client Interface'
serverinterface s	AUTHN_MECH_SET_MECHS1	0	2	

This can be accomplished using the db2 interactive utility. Here's a sample session showing the commands to use. You'll need to be logged in to Unix as user 'hpss' to have the needed permissions on the database.

```
$ db2
```

```
(c) Copyright IBM Corporation 1993,2002
Command Line Processor for DB2 SDK 8.2.5
```

```
...
```

```
For more detailed help, refer to the Online Reference Manual.
```

```
db2 => connect to hcfg
```

```
Database Connection Information
```

```
Database server          = DB2/LINUX 8.2.5
SQL authorization ID     = HPSS
Local database alias     = HCFG
```

```
db2 => set schema hpss
```

```
db2 => select num_auth_mechs, authn_mechs1_mechanism,
authn_mechs1_auth_type_key
from server where desc_name = 'SSM System Manager'
```

```
NUM_AUTH_MECHS AUTHN_MECHS1_MECHANISM AUTHN_MECHS1_AUTH_TYPE_KEY
```

```
-----
1 0 0
```

```
1 record(s) selected.
```

```
db2 => select authn_mech_set_num_mechs, authn_mech_set_mechs1 from
```



```
serverinterfaces where server_id = (select server_id from server
where desc_name = 'SSM System Manager') and descriptive_name =
'Administrative Client Interface'
```

```
AUTHN_MECH_SET_NUM_MECHS AUTHN_MECH_SET_MECHS1
-----
                                1                                0
```

1 record(s) selected.

```
db2 => update server set (num_auth_mechs, authn_mechs1_mechanism,
authn_mechs1_auth_type_key) = (2, 2, 1) where desc_name = 'SSM
System Manager'
```

DB20000I The SQL command completed successfully.

```
db2 => update serverinterfaces set (authn_mech_set_num_mechs,
authn_mech_set_mechs1) = (2, 2) where server_id = (select server_id
from server where desc_name = 'SSM System Manager') and
descriptive_name = 'Administrative Client Interface'
```

DB20000I The SQL command completed successfully.

```
db2 => select num_auth_mechs, authn_mechs1_mechanism,
authn_mechs1_auth_type_key
from server where desc_name = 'SSM System Manager'
```

```
NUM_AUTH_MECHS AUTHN_MECHS1_MECHANISM AUTHN_MECHS1_AUTH_TYPE_KEY
-----
                                2                                2                                1
```

1 record(s) selected.

```
db2 => select authn_mech_set_num_mechs, authn_mech_set_mechs1 from
serverinterfaces where server_id = (select server_id from server
where desc_name = 'SSM System Manager') and descriptive_name =
'Administrative Client Interface'
```

```
AUTHN_MECH_SET_NUM_MECHS AUTHN_MECH_SET_MECHS1
-----
                                2                                2
```

1 record(s) selected.

```
db2 => terminate
```

DB20000I The TERMINATE command completed successfully.

\$

If you're using the local HPSS password file (HPSS_UNIX_USE_SYSTEM_COMMANDS=FALSE), you need to make sure it contains entries for users 'root' and 'hpss'.

Now you can use either security mechanism ("unix" or "krb5") in your SSM configuration file.

Chapter 6. Upgrading to HPSS Release 6.2

This chapter is only intended for sites upgrading HPSS from either version 4.5 or 5.1 to version 6.2. Sites wishing to upgrade from prior HPSS releases must first upgrade to HPSS 4.5 or HPSS 5.1. Sites that are going to install and configure HPSS 6.2 from scratch will not need to perform the upgrade. The upgrade procedures in this chapter are intended only for HPSS systems running on AIX.



Due to possible risk of losing HPSS metadata, we strongly recommend that the upgrade procedure be planned and performed with the help of your HPSS customer support representative.

HPSS 6.2 replaces DCE with new authentication and authorization mechanisms. As a result, a site must decide which authentication and authorization mechanisms it wants to deploy. The steps to performing a upgrade will vary for each site depending on these choices; therefore, not all steps need to be performed by each site. Additionally, the upgrade documentation that follows is written for both 4.5 and 5.1 and some steps only apply to the release being upgraded.

The high level steps to performing the upgrade to HPSS 6.2 are outlined below:

- Verify that all prerequisite conditions are met
- Backup the current system
- Obtain, install and configure all necessary software
- Prepare the upgrade environment
- Install and configure authentication and authorization, if applicable
- Convert the metadata
- Convert authorization and authentication mechanisms
- Tune DB2
- Port prior release data such as configuration file information
- Bring up the HPSS 6.2 servers
- Verify the HPSS 6.2 system

6.1. Special Instructions for Upgrading to HPSS 6.2.2

Version HPSS 6.2.2 includes a new Tape Drive Pooling feature that requires an alteration, adding a column, to the PVLDRIVE and PVLACTIVITY tables in the HPSS global or configuration database. A new program is provided to accomplish these two changes. The program is called `hpss_tape_pool.ksh` and should be run as the DB2 instance owner or by another user with authority to create and alter, and select and insert records to/from the PVLDRIVE and PVLACTIVITY tables. The program should be run only when HPSS is down as it attempts to save the HPSS 6.2 tables in the event that reversion to HPSS 6.2 is required.

The program starts by creating two tables identical to the HPSS 6.2 PVLDRIVE and PVLACTIVITY tables called PVLDRIVE_SAVE and PVLACTIVITY_SAVE. Then the program copies all records in the PVLDRIVE and PVLACTIVITY tables into the new SAVE tables. Finally, the program alters the PVLDRIVE and PVLACTIVITY tables by adding the new DRIVE_POOL_ID column to enable the new feature and setting the initial value to 0 since it will not be enabled by default. Contact IBM Support if an error occurs or if reversion to HPSS 6.2 is needed.

6.2. Planning for the HPSS 6.2 Upgrade

This section provides information necessary to plan the upgrade of an HPSS 4.5 or HPSS 5.1 system to HPSS 6.2. It is important that all planning information be reviewed carefully before performing the upgrade.

6.2.1. Metadata changes in HPSS 6.2

From 4.5 to 6.2:

- Encina's Structured File System (SFS) product is replaced by DB2 Universal Database (UDB) as the metadata database for HPSS.
- Migration policies are now either of type disk or tape, but not both.
- The ACLs internal to an nsubject record are now placed in the NSACL table.
- The NSTEXT record is no longer used to store names longer than 23 characters. The NSOBJECT name field is now long enough to handle the entire name of the object.
- NSOBJECT records that were marked as deleted but retained for reuse are not converted.
- Disk storage algorithms are improved to eliminate previous limitations of the number of storage segments in a single virtual volume. Part of the disk storage map metadata is now managed in memory rather than in metadata.
- The Bitfile Server, Storage Server, and Name Server servers for each subsystem is now merged into a single Core Server. This has generated a number of changes in metadata, mainly in the server-specific metadata. The server-specific metadata for the Bitfile Server, Storage Server, and Name Server servers are now merged into the Core Server specific metadata.
- Cartridge metadata is now consolidated from multiple SFS files into one DB2 table.
- New AUTHZACL table. This table contains authorized interfaces to servers. The table allows a server to determine whether a client with a certain authorization mechanism (Unix or LDAP) may connect to its interface.
- Modification of DMG metadata. In support of the new HPSS RPC library: TCP port has been eliminated; Program and Version numbers to the DMG specific configuration have been added.
- Modification of DMG Fileset metadata. In support of new HPSS RPC library, elimination of TCP port and TCP hostname and addition of RPC endpoint information.
- Modification of Gatekeeper metadata. In support of new HPSS RPC library, change length of site policy pathname from 127 to 1023 characters.
- Modification of Location Server Policy metadata. In support of new HPSS RPC library, elimination of group name and addition of realm name. Change length of local site name from 31 to 255 characters.
- Modification of Mover Device metadata. In support of new SAN3P capabilities, addition of SanID to mover device configuration. Since new to HPSS 6.2, these will be set to 0 for all mover devices.
- NDCG metadata is now obsolete. The table is created, and the metadata is converted, but the table is renamed to PRE62_NDCG to prevent use.

- Modification of NFS metadata. In support of new authentication mechanisms, elimination of credential object Id. Change privileged caller principal length from 15 to 255 characters. NFS is no longer supported. The NFS table contains converted metadata, but will be renamed to PRE62_NFS to prevent use.
- Modification of server metadata. In support of new HPSS RPC library, elimination of authorization service and authentication service information. Addition of request queue size information and RPC program and version number information. In support of new HPSS authentication and authorization mechanisms, addition of authentication mechanism information.
- New SERVERINTERFACES table.
- Modification of site metadata. The conversion utilities do not convert remote site metadata because remote sites are no longer supported. The conversion will create a new empty SITE table in DB2.

From 5.1 to 6.2:

- New AUTHZACL table. This table contains authorized interfaces to servers. The table allows a server to determine whether a client with a certain authorization mechanism (Unix or LDAP) may connect to its interface. This table is populated by the `hpss_init_server_acls` conversion program.
- DISKSPACE table is no longer used in 6.2. The core server now uses in-memory disk maps rather than the metadata in this 5.1 DB2 table.
- Modification of DMG table. In support of the new HPSS RPC library: TCP port has been eliminated; Program and Version numbers to the DMG specific configuration have been added. The metadata is converted by the `hpss_51_62_dmg` conversion program.
- Modification of DMGFILESET table. In support of new HPSS RPC library, elimination of TCP port and TCP hostname and addition of RPC endpoint information. The metadata is converted by the `hpss_51_62_dmgfileset` conversion program.
- Modification of GATEKEEPER table. In support of new HPSS RPC library, change length of site policy pathname from 127 to 1023 characters. The metadata is converted by the `hpss_51_62_gatekeeper` conversion program.
- Modification of LSPOLICY table. In support of new HPSS RPC library, elimination of group name and addition of realm name. Change length of local site name from 31 to 255 characters. The metadata is converted by the `hpss_51_62_lspolicy` conversion program.
- Modification of MOVERDEVICE table. In support of new SAN3P capabilities, addition of SanID to mover device configuration. Since new to HPSS 6.2, these will be set to 0 for all mover devices. The metadata is converted by the `hpss_51_62_moverdevice` conversion program.
- NDCG table now obsolete, renamed to PRE62_NDCG to prevent use, but left intact
- NFS is no longer supported. The table is renamed to PRE62_NFS to prevent use.
- SSSTATS table now obsolete, renamed to PRE62_SSSTATS to prevent use, but left intact.
- Modification of SERVER table. In support of new HPSS RPC library, elimination of authorization service and authentication service information. Addition of request queue size information and RPC program and version number information. In support of new HPSS authentication and authorization mechanisms, addition of authentication mechanism

information. New index definition for table. The metadata conversion is performed by the `hpss_51_62_server` conversion program.

- New `SERVERINTERFACES` table. This table is populated by the `hpss_51_62_server` conversion with default server interface information.
- Modification of `SITE` table. In support of new authorization mechanism, elimination of descriptive and LS group names and addition of site and realm names. The pre-6.2 `SITE` table will be renamed to `PRE62_SITE` but the metadata in the table will not be converted into the new `SITE` table since remote sites are no longer supported.
- Modification of `STORAGESEGDISK` table. In support of new disk storage algorithms, new index definition for table.

6.2.2. Upgrade Requirements and Limitations

Prior to running any of the conversion programs, be aware of the following:

- HPSS metadata should be at the appropriate level, either HPSS 4.5 or HPSS 5.1.
- Remote site information will not be converted because remote sites are no longer supported. The conversion will create an empty HPSS 6.2 `SITE` table in DB2.
- There is no utility provided to convert the local site information from the Location Server Policy (LS Policy) into LDAP. This must be manually created with the instructions provided in Section 6.3.3: *Install and Configure LDAP* on page 191.
- There is utility provided to convert the SSM User Ids from HPSS 4.5 or 5.1 to HPSS 6.2. The `hpssuser` utility must be used to create new SSM User Ids for HPSS 6.2.
- For 4.5 upgrades only: DFS must be disabled for the duration of the upgrade.
- For 4.5 upgrades only: SFS and DB2 must be able to run simultaneously. Both databases must run on the same production server node as it is being upgraded. Therefore, SFS disk space may not be reclaimed and reused to support DB2 during the conversion process, and HPSS sites may need to install additional disks.
- For 4.5 upgrades only: A site must have only one PVL server configured. If more than one PVL is configured, the metadata conversion will be unsuccessful. An HPSS site with multiple PVLs should delete extraneous PVL servers from HPSS 4.5 prior to conversion

6.2.3. New Authentication and Authorization Mechanisms

Sites must select a new authentication and authorization mechanisms to replace DCE.

HPSS 6.2 replaces DCE by using a new HPSS RPC library, a new authentication mechanism, and a new authorization service. Sites have several options when choosing the new authentication and authorization mechanism in HPSS 6.2. Valid combinations include:

- Unix authentication and authorization with system password and group files (`/etc/passwd`, `/etc/group`). In this case, the `hpss_unix_import` program will assist in converting DCE authentication and authorization information into Unix.
- Unix authentication and authorization with HPSS password and group files (e.g. `/var/hpss/etc/passwd`, `/var/hpss/etc/group`). In this case, the `hpss_unix_import` program will assist in converting DCE authentication and authorization information into Unix.
- Kerberos authentication and LDAP authorization. In this case, the site determines on its own how to convert DCE authentication information into Kerberos. The site will use

hpss_ldap_import to convert DCE authorization information into LDAP.

- Kerberos authentication and Unix authorization. In this case, the site determines on its own how to convert DCE authentication information into Kerberos. The site will then use hpss_unix_import to convert DCE authorization information into Unix. Depending on environment variables, the hpss_unix_import program may import authentication information (i.e. Create a password for the Unix user) into Unix. The site could manually reset or remove the password from the converted Unix accounts if this is an issue after running the hpss_unix_import program.

6.2.3.1. Authentication Mechanisms

A site may select between Unix or Kerberos authentication. Some pros and cons of each are listed below.

Unix:

- Cross cell authentication is not supported.
- Can choose to use either system password or HPSS password file.
- Can degrade performance as the number of HPSS users increases due to sequential seeking through password file.
- Encryption is performed using Unix encrypt function.
- HPSS servers/processes utilize Unix keytab file.
- Can use LDAP or Unix as authorization mechanism.
- The hpss_dce_export and hpss_unix_import utilities are provided to convert DCE authentication information.

Kerberos:

- Cross cell authentication information is not converted; thus, not covered in this document.
- Using an institutional Kerberos server can complicate conversion if UID conflicts exists between current DCE principals or groups and existing Kerberos principals or groups.
- Uses underlying Kerberos encryption algorithms.
- HPSS servers/processes utilize Kerberos keytab file.
- Requires LDAP as authorization mechanism; Unix authorization not supported.
- No utilities are provided to convert DCE information to Kerberos. Site are required to perform the conversion from DCE on their own.

6.2.3.2. Authorization Mechanisms

A site may select between Unix or LDAP authorization. Some pros and cons of each are listed below.

Unix:

- Can degrade performance as the number of HPSS users increases due to sequential seeking through password file.
- Easier to setup and manage than LDAP.

- The `hpss_dce_export` and `hpss_unix_import` utilities are provided to convert DCE authorization information.

LDAP:

- Configuring LDAP is more complex than Unix.
- Managing LDAP is fairly simple and does not require regular maintenance.
- The `hpss_dce_export` and `hpss_ldap_import` utilities are provided to convert DCE authorization information.

6.2.4. New HPSS 6.2 System Files

There are several new files in `/var/hpss` that the HPSS system uses which are described in the appendix titled */var/hpss files* in Appendix E. The files located under `/var/hpss/etc` are used in providing security and communications services. Many of these files are created or modified by programs during the upgrade process. Some may require hand modifications in order to preserve the configuration settings from the previous HPSS version. Become familiar with these files before beginning the upgrade process. If problems are encountered while attempting to start the HPSS 6.2 system, the file contents should be reviewed.

6.2.5. Testing the Metadata Conversion

To minimize the downtime of a production system, we strongly recommend that sites perform a test run on a non-production system using the production metadata. The test run allows for potential issues and/or problems to be resolved prior to the metadata conversion of the production system. Additionally, the administrators will become familiar with the conversion process. Contact your HPSS customer support representative for more information on setting up a test system using the production metadata.

Sites may also practice the metadata conversion on the production system during a system maintenance period. This allows additional factors, such as system conversion load management and configuration of system resources, to be verified. However, performing this type of practice conversion requires even more detailed planning than usual to ensure that it will not interfere with the production system. Be sure to coordinate any such practice conversion with your HPSS customer support representative.

6.2.6. Estimating the Metadata Conversion Time (for 4.5 upgrades only)

The overall amount of time required to perform a metadata conversion from HPSS 4.5 depends largely on the size of the HPSS system being converted (number of files, number of file segments, number of storage segments, number of storage subsystems, etc.). The 6.2 conversion programs are separated into three categories depending on the size and type of metadata each will convert. The categories are:

- Configuration conversion
- Subsystem conversion
- Long running conversion

The time that the configuration and subsystem conversion steps take will depend mainly on the number of cartridges and the number of subsystems that the HPSS system contains. For a test HPSS system with a single subsystem containing 4,500 cartridges, the configuration conversion took about

three minutes, and the subsystem conversion took approximately 40 minutes.

6.2.6.1. Running Time for the Long Running Metadata Conversion Utilities (for 4.5 upgrades only)

A system with more than 3 million HPSS files will see a significant performance gain from choosing to run the long running metadata conversion programs in parallel. With 3 million files, the serial conversion should take approximately 11 hours to finish, while the parallel conversions would take only 2 hours.

The table below lists the estimated time required for each long running conversion program to convert one million SFS metadata records in its associated SFS file. To estimate the runtime for each conversion program, divide the number of records in the appropriate SFS file by one million and then multiply by the number given in the table. The result will be the estimated number of minutes required for that conversion program to run.

Table 10. Runing Times for Long Running Metadata Conversion Utilities

HPSS 4.5 Metadata Conversion Program	Minutes per 1 million SFS Records
db_convert_bftapeseg	24
db_convert_bitfile	20
db_convert_nsubject	38
db_convert_nstext	16
db_convert_storagesegdisk	48
db_convert_storagesegtape	48

These times will vary depending on the hardware configuration of the system on which the conversion is being run. The time estimates given above were determined on a relatively inactive system (44P-270) with about 3 million HPSS files. For example, a system in which DB2 shared one disk for all tables and table spaces (the worst configuration possible) saw performance twice as slow as what is listed above. These numbers also do not take into account the extra time it will take to run the long running metadata conversion programs in parallel.

6.2.7. Capturing the Metadata Conversion Output

All conversion output should be captured and retained for problem diagnosis and verification purposes. Prior to running a metadata conversion program, consider whether or not to run the scripts with their default behavior. Progress of the conversion programs is sent to standard out. Errors from all conversion programs are sent to standard error. We recommend that each site redirect both standard error and standard out to a file and, upon completing each conversion, search for common strings associated with problems (e.g., “Error” and “Warning”). All errors and warnings should be investigated.

Some of the conversion programs output information that allows them to restart should an error occur during a conversion run. These files are stored in the /var/hpss/convert/6.2 directory. Ensure that the filesystem containing the /var/hpss/convert/6.2 directory has at least 25 MB of free disk space prior to

running the metadata conversion. This directory will be used to store one text-based restart file, various text-based db2 output files, and other text files.

Prior to performing the metadata conversion, become familiar with the conversion program output for the HPSS 4.5 conversion in Section 6.5: *HPSS 4.5 Conversion Utilities Output* on page 232 or for the HPSS 5.1 conversion in Section 6.3.15: *Upgrade from HPSS 5.1 to HPSS 6.2* on page 213. Should any of the conversion programs fail, they will exit with an error. To correct the error and continue with the conversion, refer to Section 6.4: *Metadata Conversion Troubleshooting Procedures* on page 227.

6.2.8. DB2 Configuration and Tuning (for 4.5 upgrades only)

The DB2 database configurations, that will store the converted 6.2 metadata, must be carefully planned. In order to optimize the amount of time a conversion from HPSS 4.5 takes, it is essential that DB2 be configured specifically for HPSS. Prior to performing the conversion, verify the database configuration and make any needed adjustments. Refer to the sections below for examples on how to check and update a database configuration settings. Additionally, we recommend running several test metadata conversions to determine what DB2 settings might help or hinder performance of the conversion utilities in your specific environment. Refer to Section 6.2.5: *Testing the Metadata Conversion* on page 184. After running a metadata conversion and prior to placing the HPSS 6.2 system and DB2 databases in production, review the DB2 configuration settings that may affect the performance of the HPSS system.

For more information on how to setup and tune DB2, refer to Section 5.3.2.6: *Configure DB2 Services* on page 165 and Section 5.7: *Tune DB2* on page 171. For additional concerns about configuring DB2 that are not addressed, see the latest version of the DB2 UDB Performance Tuning Guide, an IBM redbook.

Checking Database Configuration Settings

To check database configuration setting, login as the database administrator or instance owner (e.g. hpsbdb), connect to each database and list its tables. DB2 commands are not case sensitive. To connect and list tables:

```
% DB2 CONNECT TO CFG
% DB2 LIST TABLES FOR SCHEMA HPSS
% DB2 CONNECT RESET
% DB2 CONNECT TO SUBSYS1
% DB2 LIST TABLES FOR SCHEMA HPSS
% DB2 CONNECT RESET
```

Check the database manager configuration settings at the instance level. For example:

```
% DB2 GET DBM CFG
```

Check the database configuration settings at the database level. For example:

```
% DB2 GET DBM CFG FOR CFG
% DB2 GET DBM CFG FOR SUBSYS1
```

Updating the Database Configuration

To update the database configuration settings for any databases within an instance, become the database administrator or instance owner, connect to the database and execute:

```
% DB2 UPDATE DB CFG FOR <Database Name> USING <Setting> <Value>
```

To reset to default settings, execute:

```
% DB2 RESET DB CFG FOR <Database Name>
```

Review and update the following:

- Associated with the database log are the log file size (LOGFILSIZ), the number of primary logs (LOGPRIMARY), and number of secondary log files (LOGSECOND). A good size for LOGFILSIZ prior to performing a metadata conversion is 50,000 (these are 4K pages, so each log file will be about 200 MB). LOGPRIMARY and LOGSECOND should be set to provide DB2 at least enough space for 5 times the size of the largest table. All transaction-related (insert and delete) information is handled by the log facility in DB2. Should a conversion program fail with default log size and settings, it is possible to fill the log up and run out of space (depending on the size of the table that failed). Adjust the log file size once the conversion is complete, as a log file that is too large degrades overall database performance.
- Determine if the DB2 database logging is configured properly for the system. DB2 uses the default database directory path for logging. If the database logging should utilize a separate file system than the database uses, then change the logging file system by changing the value of NEWLOGPATH. Verify the current log file system by checking the value of LOGPATH. If the DB2 logs are not going to be mirrored at the filesystem or hardware level (i.e. mirrored JFS or RAID device), we recommend that the MIRRORLOGPATH in DB2 is utilized to provide a secondary location where DB2 can store a second copy of database logs. When doing this, be sure to provide a filesystem on a separate physical device from the NEWLOGPATH or LOGPATH previously specified.
- Consider the setting of UTIL_HEAP_SZ for each subsystem database. This setting will affect the amount of memory each load operation can use and, ultimately, the number of loads that can be run in parallel. Most sites should set this to a minimum of 35,286 (4KB pages).

6.2.9. Overview of the Upgrade Utilities

This section lists the utilities that are used in the upgrade process and a brief description of each.

6.2.9.1. HPSS 4.5 and HPSS 5.1 Upgrade Utilities

The utilities listed in this section, with the exception of db_convert_dce_cds, apply to upgrades from HPSS 4.5 or 5.1 to HPSS 6.2.

Authentication/Authorization Upgrade Utilities

- db_convert_dce_cds (for 4.5 upgrades only) - The db_convert_dce_cds utility adds the DCE principal and account and sets the CDS Security ACLs for the new 6.2 Core Servers. It also updates the CDS Security ACLs for the other HPSS servers to allow access for the Core Servers
- hpss_dce_export - This utility exports principal and group information from DCE and places the information in an output file that will be used by the import programs listed below
- hpss_unix_import – This program is intended to support Unix authorization by using information from DCE about HPSS servers and HPSS users pertinent to performing authorization (UID, moniker, and home directory) and importing them into Unix system or optionally into local HPSS Unix password and group files, so that each user's specific

attributes pertinent to authorization are retained in HPSS 6.2. The program also optionally supports Unix authentication in HPSS 6.2 by obtaining each HPSS users' password from DCE and preserving it in the Unix system or optionally into a local HPSS password file

- `hpss_ldap_import` – This program is intended to support LDAP authorization by using information from DCE about HPSS servers and HPSS users pertinent to performing authorization (UID, moniker, and home directory) and importing them into LDAP
- `hpss_ldap_admin` – The `hpss_ldap_admin` utility is an LDAP administration tool that can be used to create a HPSS 6.2 site entry using the correct local site name from the Location Server Policy in HPSS 4.5 or 5.1

Server Security ACL and Endpoint Creation Utilities

- `hpss_init_server_acls` - This utility creates default server security ACLs and default interfaces for each server in HPSS 6.2. These ACLs and interfaces are necessary so that each server in HPSS 6.2 can communicate with each other
- `hpss_bld_ep` – This utility creates endpoints for the Location Server to use. The endpoints created are placed in the file `/var/hpss/etc/ep.conf`

6.2.9.2. HPSS 4.5 Upgrade Utilities

The utilities listed in this section apply only to upgrades from HPSS 4.5 to HPSS 6.2.

HPSS 4.5 to 6.2 Metadata Conversion Utilities

Prior to performing the metadata conversion, become familiar with the conversion programs and their output. Detailed information for each of these utilities is located in the sections where the utility is run. Also see Section 6.5: *HPSS 4.5 Conversion Utilities Output* on page 232 for examples of each utility's output. If the conversion program fails, it will exit with an error. To correct the error and continue with the conversion, see Section 6.4: *Metadata Conversion Troubleshooting Procedures* on page 227.

The conversion programs are located in `$HPSS_ROOT/convert62/bin`. For any of the conversion programs, simply executing the program without any options will provide the usage instructions.

The following conversion utilities are used in the 4.5 to 6.2 metadata conversion:

- `db_convert_collect_info` - The first program used in the 4.5 conversion is the information collection utility, `db_convert_collect_info`. This utility will gather configuration related information based on input provided about the 4.5 HPSS system. The program will save the information to several text files located in the `/var/hpss/convert/6.2` directory
- `db_config_convert`- The `db_config_convert` script calls a series of programs that will perform all of the HPSS 4.5 configuration metadata conversions
- `db_subsys_convert` - The `db_subsys_convert` script invokes a series of programs that will convert all of the subsystem related metadata

These 4.5 to 6.2 conversion utilities are considered "Long Running Utilities":

- `db_convert_bftapeseg` – This program converts the bitfile tape segment metadata from the HPSS 4.5 `bftapeseg` SFS file to the HPSS 6.2 DB2 `BFTAPESEG` table by reading records from SFS and performing DB2 load into the new table.
- `db_convert_bitfile` - This program converts the bitfile metadata from the HPSS 4.5 bitfile SFS file to the HPSS 6.2 DB2 `BITFILE` table by reading records from SFS and performing a DB2

load into the new table.

- `db_convert_nsubject` - This program converts the name server object metadata from the HPSS 4.5 `nsubjects` SFS file to the HPSS 6.2 DB2 `NSOBJECT` and `NSACL` tables by reading records from SFS and performing DB2 loads into the new tables.
- `db_convert_nstext` - This program converts the name server text metadata from the HPSS 4.5 `nstext` SFS file to the HPSS 6.2 DB2 `NSTEXT` table by reading records from SFS and performing a DB2 load into the new table.
- `db_convert_storagesegdisk` - This program converts the disk storage segment metadata from the HPSS 4.5 `storagesegmentdisk` SFS file to the HPSS 6.2 DB2 `STORAGESEGDISK` table by reading records from SFS and performing a DB2 load into the new table.
- `db_convert_storagesegtape` - This program converts the tape storage segment metadata from the HPSS 4.5 `storagesegmenttape` SFS file to the HPSS 6.2 DB2 `STORAGESEGTAPE` and `ABSADDR` tables by reading records from SFS and performing DB2 loads into the new tables.

HPSS 4.5 to 6.2 Verification Utilities

There are three utilities provided to verify the correctness of the HPSS 4.5 to 6.2 converted metadata.

- `db_convert_size_check` - The utility `db_convert_size_check` is provided to verify the existence and size of all the expected DB2 tables following the conversion. This utility verifies that all of the expected tables exist and compares the number of records in each table against the number of records in the corresponding SFS metadata file(s)
- `db_convert_ns_check` - The `db_convert_ns_check` utility is provided to perform a field level check of the SFS Name Server metadata against the new Core Server metadata
- `db_convert_address_check` - The `db_convert_address_check` utility is provided to take HPSS bitfile descriptors and resolve them to media volume names and addresses at each level in the hierarchy in which the file resides. The utility determines these addresses using both SFS and DB2 metadata and compares the results. Options are available to check a subset of files in the system (such as the first and last 1000 files) or every file in the system

Different degrees of verification may be performed based on the amount of time an HPSS site decides to devote to this task. Verification is highly recommended immediately following the conversion before the 6.2 HPSS servers are started.

6.2.9.3. HPSS 5.1 Upgrade Utilities

The utilities listed in this section apply only to upgrades from HPSS 5.1 to HPSS 6.2.

HPSS 5.1 to 6.2 Metadata Conversion Utilities

Prior to performing the metadata conversion, become familiar with the conversion programs and their output. Detailed information for each of these utilities is located in the sections where the utility is run. See Section 6.3.15: Upgrade from HPSS 5.1 to HPSS 6.2 on page 213 for an example of the 5.1 conversion utility output. If the conversion program fails, it will exit with an error. To correct the error and continue with the conversion, see Section 6.4: *Metadata Conversion Troubleshooting Procedures* on page 227. By running the conversion scripts successfully, a site will ensure that the DB2 database is operational and that basic database settings are correct for the configuration database.

The 5.1 to 6.2 metadata conversion is controlled by a menu driven utility called

hpss_md_convert_51. This program will rename the HPSS 5.1 tables by prepending the original table name with PRE62_ so that the original metadata in HPSS 5.1 is preserved. The hpss_md_convert_51 utility calls the following programs to perform the conversion:

- hpss_51_62_dmg – Reads the 5.1 metadata from the PRE62_DMGM table, modifies the metadata as needed, and inserts into the DMGM table for use in HPSS 6.2
- hpss_51_62_dmgfileset - Reads the 5.1 metadata from the PRE62_DMGMFILESET table, modifies the metadata as needed, and inserts into the DMGMFILESET table for use in HPSS 6.2
- hpss_51_62_gatekeeper – Reads the 5.1 metadata from the PRE62_GATEKEEPER table, modifies the metadata as needed, and inserts into the GATEKEEPER table for use in HPSS 6.2
- hpss_51_62_lspolicy - Reads the 5.1 metadata from the PRE62_LSPOLICY table, modifies the metadata as needed, and inserts into the LSPOLICY table for use in HPSS 6.2
- hpss_51_62_moverdevice - Reads the 5.1 metadata from the PRE62_MOVERDEVICE table, modifies the metadata as needed, and inserts into the MOVERDEVICE table for use in HPSS 6.2
- hpss_51_62_server - Reads the 5.1 metadata from the PRE62_DMGM table, modifies the metadata as needed, and inserts into the DMGM table for use in HPSS 6.2

HPSS 5.1 to 6.2 Verification Utilities

There are no verification utilities for the HPSS 5.1 to 6.2 conversion.

6.3. HPSS 6.2 Upgrade Procedures

This section provides the step-by-step procedures to upgrade an HPSS 4.5 or 5.1 system to HPSS 6.2. Each step must be performed in the order in which it is listed. Part of the procedure can be performed while the 4.5 or 5.1 system is running to minimize the downtime associated with the actual conversion.

The conversion procedure described in this section must be performed for each storage subsystem. Since the conversion steps vary slightly depending on whether the root subsystem or the non-root subsystem is being converted, they are identified accordingly. For clarity, the root subsystem is the subsystem which owns the HPSS "/" directory, usually subsystem 1. A non-root subsystem is any other subsystem in the HPSS system.

- *Some steps in this section can be performed while the HPSS 4.5 or 5.1 system is running*



- *All conversion steps must be performed while running as root*

6.3.1. Verify Prerequisites

Verify that all conversion requirements are met and any limitation issues are resolved. Refer to Section 6.2.2: Upgrade Requirements and Limitations on page 182 for more information.

6.3.2. Acquire Software

Refer to Section 3.3.2, *Prerequisite Summary By HPSS Node Type* on page 59 for the appropriate software levels.

- Acquire Kerberos or LDAP software, as needed. See Section 6.2.3: *New Authentication and Authorization Mechanisms* on page 182 for a list of valid authentication and authorization combinations. See Section 5.2: *Install Prerequisite Software* on page 137 for more information on obtaining MIT or IBM Kerberos and LDAP.
- Acquire DB2 UDB
- Acquire software to upgrade AIX, if necessary
- Acquire Java software
- Acquire HPSS Release 6.2 distribution images

6.3.3. Install Authentication and Authorization Mechanisms

Select the desired authentication and authorization mechanisms to replace DCE. See Section 6.2.3: *New Authentication and Authorization Mechanisms* on page 182 to aid in choosing a mechanism. See Section 5.2: *Install Prerequisite Software* on page 137 for more information on installing MIT or IBM Kerberos or LDAP.

Unix Authentication and Authorization

If Unix is selected for the authentication and authorization method, no special instructions are required for this step.

Install and Configure Kerberos

- *Kerberos must be installed in order to successfully compile and use the PFTP client with HPSS 6.2.*
- *This step can be performed while the HPSS 4.5 or 5.1 system is running.*

Configuration of Kerberos will not be covered in this document, as converting principals and groups from DCE into Kerberos is handled outside of the conversion process. If Kerberos authentication is selected, the site is responsible for ensuring that DCE account information (principal, group, password) is transferred into Kerberos by their own means.

Install and Configure LDAP

- *The steps in this section can be performed while the HPSS 4.5 or 5.1 system is running.*

LDAP requires 400MB free space in /opt/IBM/ldap/V6.0. To install LDAP, untar the LDAP release file (e.g. itds60-aix-ppc-native.tar). After the LDAP code is extracted, use smitty or the software installation tool of choice to install the code/packages. The steps described below will assist a site with setting up LDAP with simple authentication rather than with Kerberos authentication.

5. Ensure the ldap user and ldap group exist. Create the hpsldap user and add to the HPSS DB2 INSTANCE_OWNER group (e.g. hpsbdb). Ensure root is in the instance owner group as well. Perform a login command after creating the hpsldap user to initialize the password for the new user.
6. Add the db2profile lines to the new LDAP instance owner's .profile or .cshrc. See the HPSS instance owner's .profile for an example. Ensure the db2profile is sourced for the LDAP

instance (e.g. `./home/hpssldap/sqllib/db2profile`)

7. Setup DB2 instance for LDAP:

```
% /opt/IBM/ldap/V6.0/sbin/idsicrt -n -e <random string> -I  
<ldap instance name> -t <user name> -l <dir for ldap instance>
```

e.g.

```
% /opt/IBM/ldap/V6.0/sbin/idsicrt -n -e [asUf4As].f -I idsldap  
-t idsldap -l /home/idsldap
```

8. Set the LDAP admin distinguished name (DN) and password by:

```
% /opt/IBM/ldap/V6.0/sbin/idsdnpw -I <ldap instance name> -u  
<distinguished name> -p <password>
```

e.g.

```
% /opt/IBM/ldap/V6.0/sbin/idsdnpw -I idsldap -u "cn=hpssldap"  
-p hpssldap
```

9. Add suffix for realm DN:

```
% /opt/IBM/ldap/V6.0/sbin/idscfgsuf -I <ldap instance name> -s  
<realm name>
```

e.g.

```
% /opt/IBM/ldap/V6.0/sbin/idscfgsuf -I idsldap -s  
'cn=hpss.acme.com'
```

10. Create the database user. For example, create ldapdb2 user for database, set password to ldapdb2, and make primary group idsldap. Initialize the password for this user.

11. Start the LDAP server:

```
% idsslapd -I <ldap instance name>
```

e.g.

```
% idsslapd -I idsldap
```

12. Create and configure the HPSS LDAP database:

```
% /opt/IBM/ldap/V6.0/sbin/idscfgdb -I <ldap instance name> -a  
<database user> -w <user password> -t <database name> -l  
<database dir>
```

e.g.

```
% /opt/IBM/ldap/V6.0/sbin/idscfgdb -I idsldap -a ldapdb2 -w  
ldapdb2 -t ldapdb2 -l /home/ldapdb2
```

13. Import the HPSS LDAP schema to the newly created database:


```
% /opt/IBM/ldap/V6.0/bin/idsldapmodify -c -i /opt/hpss/config/
HPSS_ldap_schema.ldif -D "cn=hpssldap" -w <password> -v
```

Note: where passwd is the password for the DN. In the example above, it would be the hpssldap user's password since hpssldap is the DN used above and configured with LDAP on this system.

14. Setup LDAP stash file in /var/hpss/etc/ldap_stash:

```
% cat /var/hpss/etc/ldap_stash

ldap {
    host=hpss.acme.com
    dn=cn=hpssldap
    password=<password>
}
```

Note: <password> refers to hpssldap's password on the system.

6.3.4. Install or Upgrade DB2

Refer to Section 3.3.2: *Prerequisite Summary By HPSS Node Type* on page 59 for the appropriate version and level of DB2.

Install and Configure DB2 (for 4.5 upgrades only)



- *This step can be performed while the HPSS 4.5 system is running*
- *It is very important to correctly specify whether the database is being created for a root or non-root subsystem so that the appropriate database(s) can be created*
- *Use the planning information from Section : on page to provide input for the mkhpss script*

These instructions should only be followed for a site upgrading from HPSS 4.5 that does not already have DB2 installed or configured on their system. A site should follow the instructions in Section 5.3.1.2: *Install HPSS Documentation and DB2 Software* on page 141 and Section 5.3.2.6: *Configure DB2 Services* on page 165 to install and configure DB2. Before proceeding past this step, verify that all the necessary tables and indexes exist for a 6.2 system. Perform this step for each root and non-root subsystem.

Following installation and configuration, tune the DB2 configuration in preparation for the conversion. Refer to Section 5.7: *Tune DB2* on page 171 for more information.

Upgrading DB2 (for 5.1 upgrades only)



This step cannot be performed while the HPSS 5.1 system is running.

This section only applies to sites that are upgrading from HPSS 5.1 to HPSS 6.2. There are no special considerations for HPSS in upgrading DB2. Each site should follow the FixPak documentation that is provided with the FixPak for instructions on how to upgrade DB2. Perform this step for each root and non-root subsystem.

6.3.5. Upgrade AIX



This step cannot be performed while the HPSS 4.5 or 5.1 system is running.

This section applies to site that are upgrading to HPSS 6.2 from both HPSS 4.5 and HPSS 5.1. See Section 3.3.2: *Prerequisite Summary By HPSS Node Type* on page 59 for the the appropriate version of AIX required for HPSS 6.2. This step should be performed on all systems used with HPSS 6.2.

6.3.6. Install or Upgrade Java



This step can be performed while the HPSS 4.5 or 5.1 system is running.

See Section 3.3.2: *Prerequisite Summary By HPSS Node Type* on page 59 for the the appropriate version of Java required for HPSS 6.2. Perform this step for the root subsystem only.

6.3.7. Save Current HPSS Code and Configuration Files



This step should not be performed while the HPSS 4.5 or 5.1 system is running.

After the prerequisite software has been installed, perform the following for each root and non-root subsystem:

1. Logon as root user

```
% su -
```
2. Save the installed HPSS 4.5 or 5.1 code, *where xx = 45 or 51*.

```
% cd /opt
% mv hpss hpss_XX
% mkdir hpss
```
3. Save the HPSS 4.5 or 5.1 /var/hpss directory and its contents, *where xx = 45 or 51*.

```
% cd /var
% cp -pR hpss hpss_XX
```
4. Copy the 4.5 or 5.1 mkhpss' state files, if they exist

```
% cd /var/hpss
% cp -p /opt/hpss_XX/config/mkhpss.data* .
% cp -p /opt/hpss_XX/config/hpss_unmake* .
```

6.3.8. Prepare HPSS 6.2 Code

Install and if necessary compile the HPSS 6.2 distribution image.

6.3.8.1. Install HPSS 6.2 Distribution Image



This step can be performed while the HPSS 4.5 or 5.1 system is running.

Prior to installing the HPSS 6.2 code, sites should not disturb the current HPSS release code. It is possible for a site to revert to the previous release of HPSS at any point prior to the last step of starting the HPSS 6.2 servers. In order to facilitate reverting to the previous release of HPSS, sites are recommended to install the HPSS 6.2 code in a separate file system from the previous release of HPSS code and set up BUILD_TOP_ROOT as a symbolic link. For example, if HPSS 4.5 or 5.1 code is installed in /opt/hpss, a site could install HPSS 6.2 in /opt/hpss62 and make /opt/hpss as the symbolic link pointing to the version of HPSS they desire to run. To proceed with this conversion, BUILD_TOP_ROOT should point the symbolic link, and the symbolic link should point to the HPSS 6.2 code.

6.3.8.2. Compile HPSS 6.2 Source Code (if necessary)

This section applies to sites that are upgrading from both HPSS 4.5 and HPSS 5.1. Install the HPSS 6.2 distribution images. Edit Makefile.macros and the Makefile.macros.* for the specific platform being used (e.g. Makefile.macros.AIX).

Constants that normally need modification in Makefile.macros include:

- BUILD_PLATFORM
- BUILD_TOP_ROOT
- BUILD_UNSUPPORTED
- CONVERSION_SUPPORT
- CONVERSION_FROM_51
- CONVERSION_FROM_45
- MVR options (MVR1_OPTION)

Note that BUILD_TOP_ROOT should point to the HPSS 6.2 code. CONVERSION_SUPPORT should be set to on and either CONVERSION_FROM_51 or CONVERSION_FROM_45 should be on depending on the version of HPSS a site is upgrading from. If CONVERSION_FROM_45 is on, the site must have Encina SFS installed and should ensure that the ENCINA constant in Makefile.macros is correct.

Constants normally needing modification in Makefile.macros.AIX include:

- DB_INSTALL_PATH
- LDAP_INSTALL_PATH – only if using LDAP authorization
- KRB5PATH – only if using Kerberos authentication
- STK_INCLUDE
- STK_LIB
- PVR_LIST

After updating Makefile.macros and the platform specific Makefile.macros, compile the HPSS 6.2

code.

Check \$HPSS_ROOT/bin/convert62 to ensure the conversion programs compiled properly. The conversion utilities must include several Encina SFS files to compile and link using /usr/lpp/encina as the default pathname. If this problem is encountered during building the source code, create a link from /usr/lpp/encina to where the root Encina SFS directory resides.

6.3.8.3. Disable Binaries, temporarily



Ensure that migration, purge, repack, reclaim, and shelf tape operations are not performed until files are read/written successfully to minimize user data corruptions, if the 6.2 system has to be reverted.

Remove all execution permissions for the following binaries in the /opt/hpss/bin directory to prevent accidental invocations until the HPSS 6.2 system has been completely verified. Perform this step on each root and non-root subsystem.

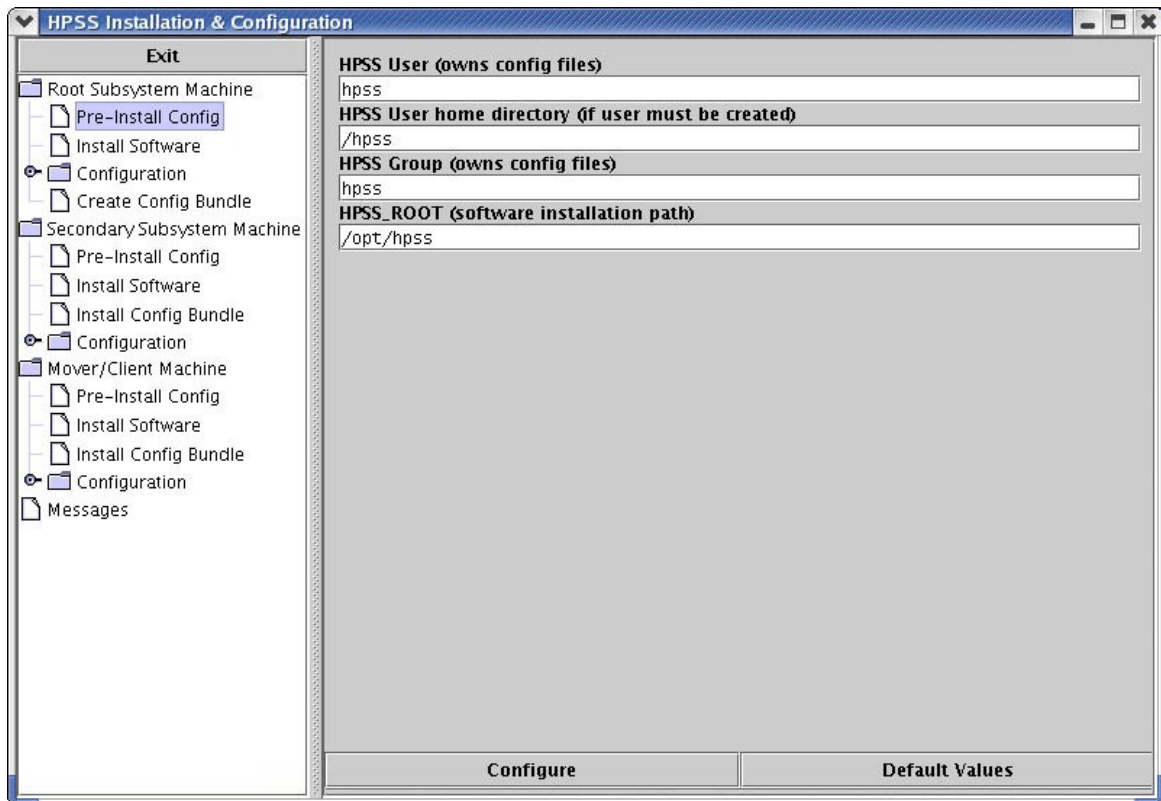
- hpss_mps
- repack
- reclaim
- recover
- shelf_tape
- shelf_tape_util

6.3.9. Set Environment Variables

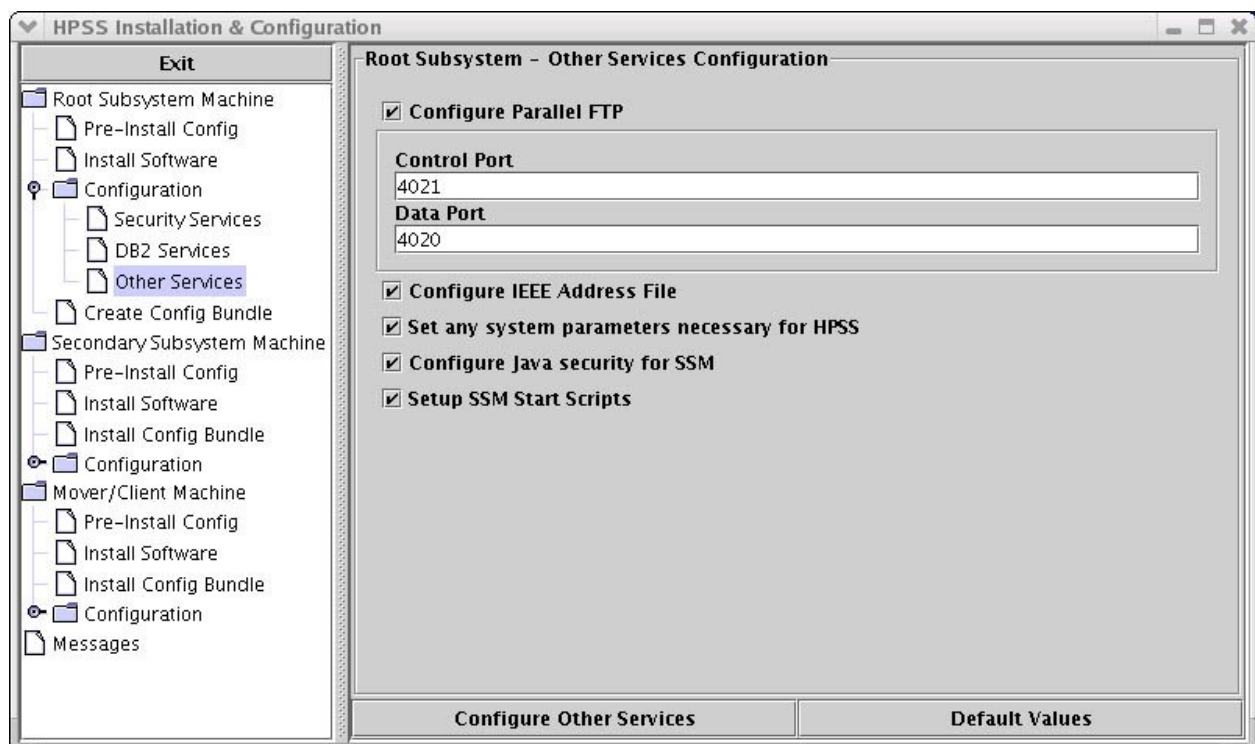
This section applies to sites that are upgrading from both HPSS 4.5 and HPSS 5.1.

In preparation for both the HPSS 4.5 and HPSS 5.1 conversion to HPSS 6.2, run the mkhpss utility and select the Pre-Install Config option according to the type of machine desired (e.g. Root Subsystem Machine) to setup the HPSS 6.2 environment. See the screen shot below for an example:

```
% /opt/hpss/bin/mkhpss
```



Next, use `mkhps` and select the Configuration option and then Other Services option to configure the IEEE address for HPSS. This will create the `/var/hpss/etc/ieee_802_addr` file. This is only used by HPSS and will not affect other services on this system. Without performing this option the HPSS RPC mechanisms will fail. Note that an error may occur about not finding the SSM server configuration, and this is expected and should be ignored because the SSM server metadata has not been converted yet. For an example, see the screen shot below:



Environment variables are no longer kept in the \$HPSS_ROOT/config subdirectory. Many are now kept in /var/hpss/etc/env.conf. The mkhpss utility will assist in creating the env.conf file and HPSS specific environment variables can be added to it.

Before running the conversion utilities, ensure the following environment variables are properly defined. These environment variables can be defined in the new env.conf file or any other way a site defines environment variables. Ensure the HPSS server principal names below are set to proper HPSS 6.2 values to match the restrictions of the chosen authentication and authorization mechanism. For instance, if Unix authentication or authorization is chosen, each principal should be no longer than 8 characters. The conversion uses these values to determine how to properly convert the system:

- DB2INSTANCE – the Unix User ID of the DB2 HPSS instance (e.g. "hpssdb"). The rc.db2 script sets this environment variable
- HPSS_GLOBAL_DB_NAME – the database name for the HPSS global configuration database (e.g. "cfg"). Defined in /var/hpss/etc/env.conf. Used to recommend default values for the current DB2 configuration
- HPSS_MM_SCHEMA_NAME – the schema for HPSS (e.g. "hpss"). Defined in /var/hpss/etc/env.conf. Used to recommend default values for the current DB2 configuration
- HPSS_SEC_REALM_NAME – the Kerberos realm name to use (e.g. "hpss.acme.com"). Defined in the /var/hpss/etc/env.conf file
- DB_USER – set to HPSS DB2 instance owner user name (e.g. "hpssdb")
- DB_PASSWORD – set to DB_USER's password
- HPSS_ROOT – set to root directory of HPSS 6.2 code (e.g. "/opt/hpss"). The rc.db2 script sets this environment variable
- HPSS_SERVER_DB_GROUP – set to the new group name for HPSS servers (e.g. "hpsssvr")
- JAVA_ROOT – set to the path to Java 1.4, platform specific (e.g. "/usr/java14")

- MKHPSS_ROOT – set to the path of HPSS 6.2 code (e.g. "/opt/hpss")
- HPSS_DB_INSTANCE_OWNER – the Unix User ID of the DB2 HPSS instance (e.g. "hpssdb")
- HPSS_LDAP_BIND_TYPE – should correspond with the type of bindings allowed for LDAP (e.g. "SIMPLE")
- HPSS_LDAP_URL – the valid URL for the HPSS LDAP server (e.g. "ldap://hpss.acme.com/cn=hpss.acme.com")
- HPSS_LDAP_BIND_ARG – the name of the file created previously to hold the user name and password of the LDAP owner (e.g. "/var/hpss/etc/ldap_stash")
- HPSS_LDAP_HOST – the host name of the LDAP server (e.g. "hpss.acme.com")
- HPSS_LDAP_REALM – the realm name for HPSS on the LDAP server (e.g. "hpss.acme.com")
- HPSS_PRINCIPAL_CORE – the 6.2 principal name for the Core Server (e.g. hpsscore)
- HPSS_PRINCIPAL_DMG – the 6.2 principal name for the DMAP Gateway server (e.g. hpssdmg)
- HPSS_PRINCIPAL_FTPD – the 6.2 principal name for the FTP Daemon (e.g. hpssftp)
- HPSS_PRINCIPAL_GK – the 6.2 principal name for the Gatekeeper (e.g. hpssgk)
- HPSS_PRINCIPAL_HPSSD – the 6.2 principal name for the Startup Daemon (e.g. hpsssd)
- HPSS_PRINCIPAL_LOG – the 6.2 principal for the Log Client and Log Daemon (e.g. hpsslog)
- HPSS_PRINCIPAL_LS – the 6.2 principal for the Location Server (e.g. hpssls)
- HPSS_PRINCIPAL_MOUNTD – the 6.2 principal for the Mount Daemon (e.g. hpssmtd)
- HPSS_PRINCIPAL_MPS – the 6.2 principal for the Migration/Purge Server (e.g. hpssmps)
- HPSS_PRINCIPAL_MVR – the 6.2 principal for the Mover (e.g. hpssmvr)
- HPSS_PRINCIPAL_PVL – the 6.2 principal for the Physical Volume Library server (e.g. hpsspvl)
- HPSS_PRINCIPAL_PVR – the 6.2 principal for the Physical Volume Repository server (e.g. hpsspvr)
- HPSS_PRINCIPAL_SSM – the 6.2 principal for the Storage System Manager (e.g. hpsssm)

6.3.10. Setup Authentication and Authorization

The mkhpss utility is used to setup authentication and authorization for the HPSS 6.2 system. See Section 6.2.3: *New Authentication and Authorization Mechanisms* page 182 for a list of the supported authentication and authorization mechanisms. In HPSS 6.2, the new authentication mechanism used by each server is determined by the environment variables HPSS_PRIMARY_AUTHN_MECH and HPSS_PRIMARY_AUTHENTICATOR. These environment variables will be set in the /var/hpss/etc/env.conf file.

Valid settings for Kerberos authentication are:

```
HPSS_PRIMARY_AUTHN_MECH=krb5
HPSS_PRIMARY_AUTHENTICATOR=/krb5/hpss.keytabs
```

Valid settings for unix authentication are:

```
HPSS_PRIMARY_AUTHN_MECH=unix
HPSS_PRIMARY_AUTHENTICATOR=/etc/passwd
```

The new authorization is performed based on the contents of `/var/hpss/etc/site.conf`. This file will tell HPSS to use unix authorization, or to use LDAP and how to contact the LDAP server based on the URL entered.

Invoke the `mkhps` utility and select the appropriate options to support the authentication and authorization mechanisms chosen previously.

```
% /opt/hpss/bin/mkhps
```

There are four main sections to this screen:

- Configure Authentication Service
- Configure Authorization Service
- Enable Local Unix Passwd Files
- Configure server accounts

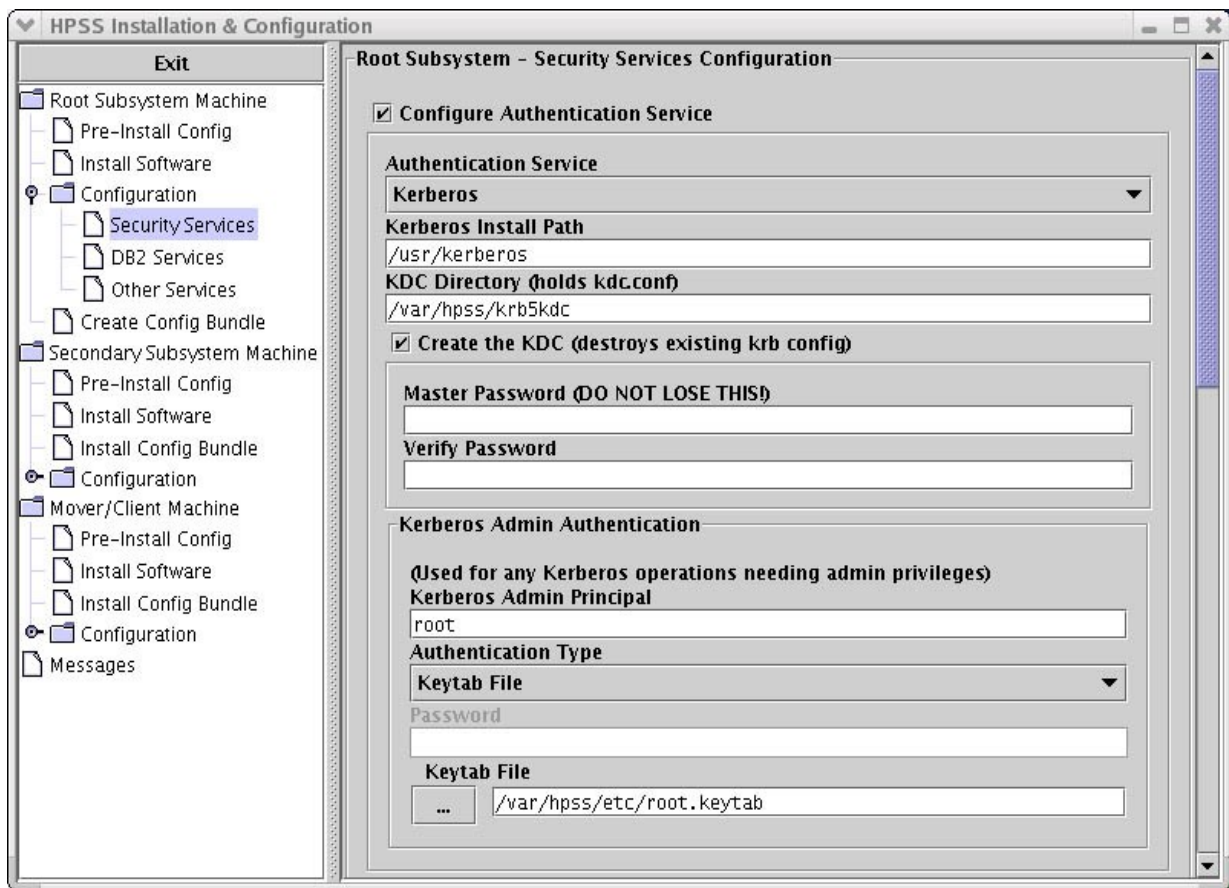
In the "Configure Authentication Service" section, set the Authentication Service field to Kerberos or Unix by means of the drop down menu on the field. If Unix is selected, no further modification to this section is necessary. If Kerberos is selected, complete the remaining fields of the section appropriately. Be certain to unselect the "Create the KDC" subsection if your KDC already exists and you do not want `mkhps` to recreate it.

In the "Configure Authorization Service" section, set the Local Site Name, Local Realm Name, and Local Realm ID. Set the Authorization Service field to "Unix and config files" or "LDAP" by means of the drop down menu on the field. If Unix is selected, no further modification to this section is necessary. If LDAP is selected, complete the remaining fields of the section appropriately.

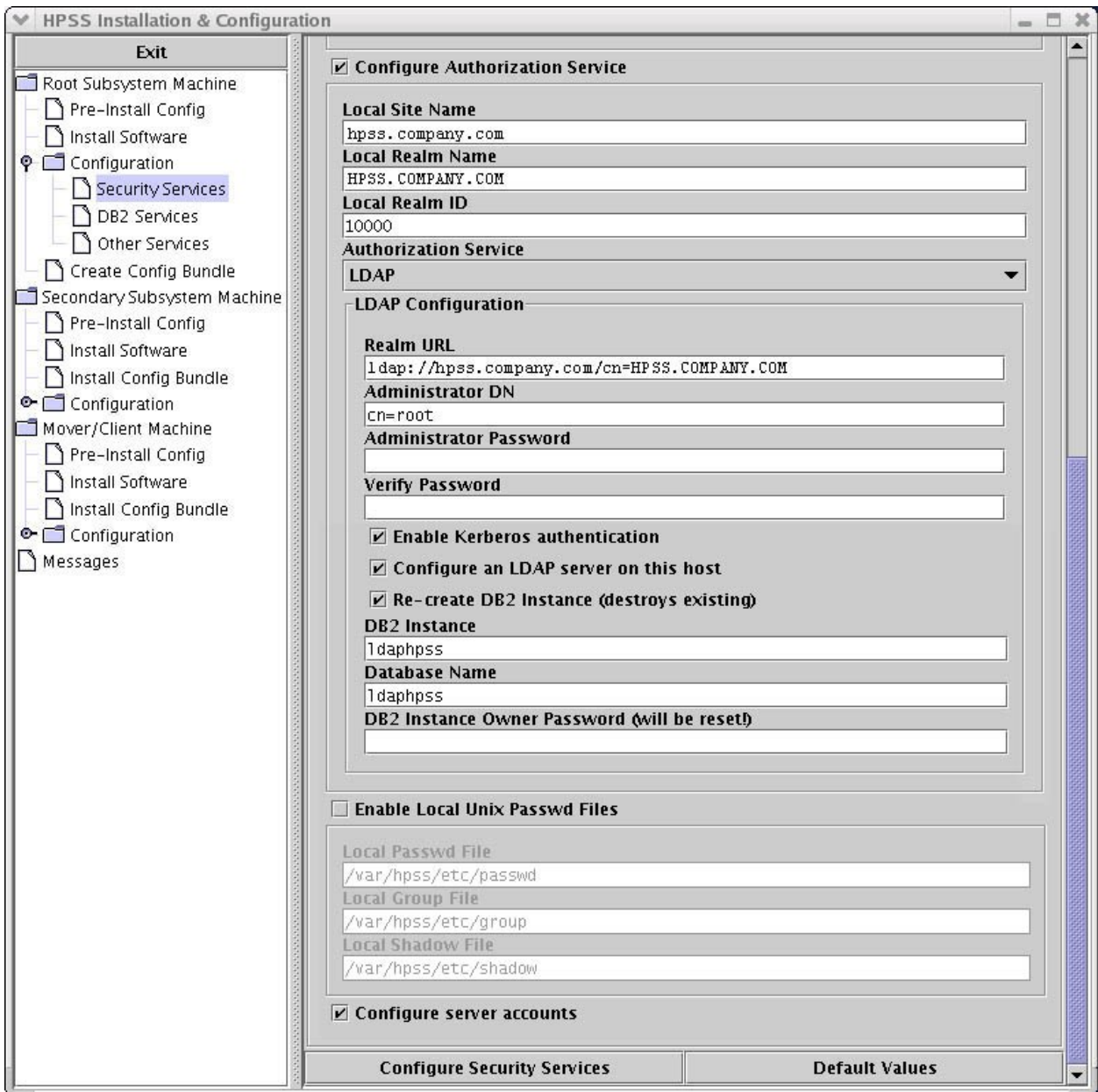
Enable the checkbox of the "Enable Local Unix Passwd Files" section and complete all the fields if you want to use a set of password and group files for HPSS use only. If you wish to use system password and group files (such as `/etc/passwd` and `/etc/group`, for example), unselect the checkbox for this section.

Enable the checkbox for the "Configure server accounts" section to request `mkhps` to create accounts for the HPSS server principals.

An example configuration for a site that desires Kerberos authentication with LDAP authorization is displayed below:



Using the scrollbar, move the right-panel scrollbar until the "Authorization Service" information is seen. It should look like the following:



6.3.11. Pre-Conversion System Check



- *It is strongly recommended that a test conversion be performed prior to running the conversion on a production system. See Section 184: Testing the Metadata Conversion on page 184*
- *Ensure that the DB2 database has been configured prior to the conversion*
- *Conversion output should be captured and any problems corrected before continuing. See Section 6.2.7: Capturing the Metadata Conversion Output on page 185 for information on capturing and viewing the conversion output*

See Section 6.2.6: *Estimating the Metadata Conversion Time* on page 184 to estimate the running time for the HPSS 4.5 conversion utilities. See Section 5.7: *Tune DB2* on page 171 for DB2 Tuning needed prior to performing the conversions (for 4.5 upgrades only).

6.3.12. Take a full backup of SFS or DB2



This step should not be performed while the HPSS 4.5 or 5.1 system is running.

The SFS or DB2 backup should be performed on the root and each non-root subsystem. Refer to the HPSS 4.5 or 5.1 Management Guide or contact your HPSS customer support representative for instructions on how to complete a SFS or DB2 backup.

6.3.13. Upgrade from HPSS 4.5 to HPSS 6.2

Only follow procedures in this section to convert an HPSS 4.5 system to HPSS 6.2. The conversion programs do not modify HPSS 4.5 metadata so that it is possible to revert to the HPSS 4.5 system during any point in the conversion process. Run all conversion programs as user root and ensure that root has “.” in it’s PATH. All conversion programs are located in HPSS_ROOT/bin/convert62.



- *The HPSS 4.5 system must be down while the following steps are performed. Note that all 4.5 SFS server(s) must be up and running while the 4.5 metadata is being converted*
- *It is also mandatory that the convert steps be performed in the order shown. However, if the long running conversion utilities are run in parallel mode, the utilities themselves can be invoked in any order as long as they are run as part of the long running conversion step*

The original SFS database is not altered in any way by the conversion. It is possible to revert to the 4.5 HPSS code/binaries and the SFS database at any point, until HPSS is started on the 6.2 database. See Section 6.2.7: Capturing the Metadata Conversion Output on page 185. If errors occur, see Section 6.4: Metadata Conversion Troubleshooting Procedures on page 227 before contacting your HPSS customer support representative. Prior to running the scripts, ensure SFS and DB2 are running.

Perform the following steps for each root and non-root storage subsystem:

6.3.13.1. Prepare for the Conversion

- Shutdown all 4.5 HPSS servers but leave the SFS server(s) running
- Disable DFS including read-only mode
- Login as root

```
% su - root
```

- Set up scripting session to capture conversion output

```
% cd /opt/hpss/bin/convert62  
% script convert.output
```

6.3.13.2. Run db_convert_collect_info to Collect Metadata Information

Run the db_convert_collect_info script.

```
% db_convert_collect_info /./encina/sfs/hpss/globalconfig
```

The first program used in the 4.5 conversion is the information collection utility, `db_convert_collect_info`. This utility will gather configuration related information based on input provided about the 4.5 HPSS system. The program will save the information to several text files located in the `/var/hpss/convert/6.2` directory. This utility must be run once for each subsystem and must be run prior to invoking any other conversion utilities.

All configuration metadata will be placed in the global configuration database designated when running the `db_convert_collect_info` utility. Subsystem metadata should be placed in its own separate database. When prompted by the `db_convert_collect_info` utility for a subsystem database name, ensure a unique database name is provided for each subsystem.

6.3.13.3. Convert Configuration Metadata

Run the `db_config_convert` script. Perform this step on the root subsystem only.

```
% db_config_convert ././encina/sfs/hpss/globalconfig
```

This script calls a series of programs that will perform all of the HPSS 4.5 configuration metadata conversions. This script provides a good way of performing a practice conversion. The program runs in a matter of minutes depending on the complexity of the the 4.5 HPSS system (i.e. the number of subsystems and the total number of servers), and can run against a fully active 4.5 HPSS system.

Prior to running this script, execute the `db_convert_collect_info` script and ensure both SFS and DB2 are running. This script only needs to run once on the root subsystem node. To run the script, login as root and provide the complete and correct SFS global configuration file CDS pathname as a command line argument. For example,

```
% su - root
% db_config_convert ././encina/sfs/hpss/globalconfig
```

This script has no restart capability, but because it is short running and it performs read-only operations on the 4.5 metadata, it can be rerun again if desired or if it encounters an error. When running this script a second time and if you want to delete the previously converted metadata, answer 'Y' when prompted to empty the related DB2 tables that were populated during the previous run; otherwise, the script will terminate due to an error (-2014) related to duplicate DB2 entries. The script places all configuration metadata into the global "cfg" database.

To determine whether the program completed successfully, search the output file for keywords such as "Error:" and "Warning:". See Section 6.2.7: *Capturing the Metadata Conversion Output* on page 185 and Section 6.4: *Metadata Conversion Troubleshooting Procedures* on page 227.

6.3.13.4. Convert Subsystem Metadata

Run the `db_subsys_convert` script.

```
% db_subsys_convert ././encina/sfs/hpss/globalconfig <subsys id>
[readonly]
```

The `db_subsys_convert` script invokes a series of programs that will convert all of the subsystem related metadata. The script completes execution in less than an hour depending on the complexity or size of your HPSS system (i.e. the number of subsystems and the total number of servers) and can run against a fully active 4.5 HPSS system.

Prior to running this script, execute the `db_convert_collect_info` script and ensure both SFS and DB2 are running. This script must be run once for each subsystem. To run the script, first login as root. Next, provide the complete and correct SFS global configuration file CDS pathname and the

subsystem Id of the subsystem being converted as the command line arguments. For example,

```
% su - root
% db_subsys_convert /./encina/sfs/hpss/globalconfig 1
```

This program provides an option to change all partially written HPSS 4.5 tapes to be read-only. When invoked, the option will force the storage map tape conversion to look at the current state (administrative and map states associated with storage map tape and tape virtual volume metadata) of each tape volume in HPSS 4.5 and, if the cartridge is determined to have data on it and be writable, it will set the state of the virtual volume to be non-writable. This will prevent further data from being written to any partially written tape from HPSS 4.5 after the conversion. This allows sites to migrate all data from disk to tape prior to a conversion, invoke the 'readonly' option on this program, and ensure that a reversion to HPSS 4.5 can be achieved without loss of user data even after writing to disk and tape with the converted HPSS 6.2 system.

This script has no restart capability, but because it is short running and it performs read-only operations on the 4.5 metadata, it can be rerun again if desired or if it encounters an error. When rerunning this script a second time and if you want to delete the previously converted metadata, answer 'Y' when prompted to empty the related DB2 tables that were populated during the previous run; otherwise, the script will terminate due to an error (-2014) related to duplicate DB2 entries. The script places all configuration metadata into the global configuration database.

To determine whether the program completed successfully, search the output file for keywords such as "Error:" and "Warning:". See Section 6.2.7: Capturing the Metadata Conversion Output on page 185 and Section 6.4: *Metadata Conversion Troubleshooting Procedures* on page 227.

6.3.13.5. Run the Long Running Utilities

The long running utilities include `db_convert_bftapeseg`, `db_convert_bitfile`, `db_convert_nsubject`, `db_convert_nstext`, `db_convert_storagesegdisk`, and `db_convert_storagesegtape`. The long running conversion utilities have restart capability, meaning that if the programs are run subsequent times with the `-restart` option, they will be able to pick-up where they left off in the previous run. Instructions on how to perform the restart is described below.



*Ensure that the node running the conversion has adequate system resources such as memory, paging space, etc. to handle the load required by the concurrent long running conversion utilities. If there are not sufficient resources, the conversion(s) may fail. If running the individual long running programs in parallel, the amount of memory allocated to the load operations within DB2 is an important factor to consider. Consider adjusting the database setting `UTIL_HEAP_SZ` for each subsystem database temporarily to handle the loads in parallel or consider running a limited number of conversions in parallel at a time. Other DB2 configuration settings may also need to be adjusted. Refer to Section 5.7: *Tune DB2* on page 171 for more information.*

Run the scripts as user root. If overall conversion time or performance is important, then run the long running conversion programs in parallel by invoking them individually and concurrently as shown in example below. Ensure there is a space between each command line option or flag and its argument. The six long running conversion utilities can be invoked in any order. Each program will prompt the site before proceeding with the conversion as to whether to empty the tables and the metadata in DB2; answer 'y' to continue.

```
% su - root
% db_convert_bftapeseg -g /./encina/sfs/hpss/globalconfig \
```

```

        -db /var/hpss/convert/6.2/Convert.DB.Names \
        -ss /var/hpss/convert/6.2/Convert.SS.Server.Ids \
        -sub 1

% db_convert_bitfile -g /./encina/sfs/hpss/globalconfig \
        -db /var/hpss/convert/6.2/Convert.DB.Names \
        -sub 1

% db_convert_nsubject -g /./encina/sfs/hpss/globalconfig \
        -db /var/hpss/convert/6.2/Convert.DB.Names \
        -ns /var/hpss/convert/6.2/Convert.NS.Server.Ids \
        -sub 1

% db_convert_nstext -g /./encina/sfs/hpss/globalconfig \
        -db /var/hpss/convert/6.2/Convert.DB.Names \
        -sub 1

% db_convert_storagesegdisk -g /./encina/sfs/hpss/globalconfig \
        -db /var/hpss/convert/6.2/Convert.DB.Names \
        -ss /var/hpss/convert/6.2/Convert.SS.Server.Ids \
        -sub 1

% db_convert_storagesegtape -g /./encina/sfs/hpss/globalconfig \
        -db /var/hpss/convert/6.2/Convert.DB.Names \
        -ss /var/hpss/convert/6.2/Convert.SS.Server.Ids \
        -sub 1

```

In the above examples, the large tables are converted for subsystem 1. If the 4.5 HPSS system has more than one subsystem, run each of these programs in parallel for each subsystem to convert.

It is important to note that the long running conversion programs could take a significant amount of time to complete, and consider executing each of these programs with the `nohup` command. To prevent accidental loss of data, each long running conversion program will prompt for execution confirmation when first starting the program, as each of these programs empties the appropriate DB2 table (i.e. deletes metadata) at the beginning of its execution. To override the prompting, provide the `-noprompt` flag to each program on the command line. When using `nohup`, supply the `-noprompt` flag to prevent the program from hanging.

Here is an example of starting a long running conversion using the `nohup` command:

```

% nohup db_convert_nsubject -g /./encina/sfs/hpss/globalconfig \
        -db /var/hpss/convert/6.2/Convert.DB.Names \
        -ns /var/hpss/convert/6.2/Convert.NS.Server.Ids -sub 1 \
        -noprompt > nsubject.out 2>&1 &

```

To be able to properly restart any of the long running conversion programs, it is imperative that each long running conversion program is only terminated by sending the appropriate C program a `SIGTERM` or `SIGINT` signal. For example, to stop the bitfile conversion program, first determine the PID of the bitfile C program. To do this, issue:

```

% ps -eaf | grep bitfile

```

This would provide the following output:

```

root 1381 1845 0 16:04:05 pts/13 0:00 lr_db_convert_bitfile ...

```

```
root 1456 2034 0 16:04:08 pts/13 0:00 db_convert_bitfile ...
root 1567 1456 0 16:04:08 pts/13 0:00 ksh db2 load ...
```

Next, issue a 'kill -15 1381', sending a SIGTERM (kill -l, will list all signals and their appropriate number on your system) to the only the conversion C program, and not to any of the other bitfile conversion processes. Note that all C conversion programs begin with lr_db_convert. This will allow the C program to flush its buffers and write a checkpoint record to a file for use by the conversion programs upon using the -restart option. If the improper signal (i.e. KILL) is sent to any of the conversion programs, they may not be able to restart properly. We recommend that you contact your HPSS customer support representative for assistance.

Most long running conversions perform a load/replace operation, which means they will automatically empty or throw away all records (metadata) in the tables they are loading data into from SFS. These programs include:

- db_convert_bftapeseg
- db_convert_bitfile
- db_convert_nstext
- db_convert_storagesegdisk

If needed, rerun the above programs in any order because they always replace the records in the tables they are loading and do not depend on any other conversions.

Other long running conversions perform inserts on certain tables, which means that they will never replace or destroy records (metadata) that exist in the tables when they are run. This is because they depend on other conversion programs (db_convert_nsac1 and db_convert_vvtape) that are part of the subsystem conversion program (db_subsys_convert). Because these long running programs perform load insert operations, consider explicitly deleting the records (metadata) from the ABSADDR and NSACL tables in order to prevent errors about duplicate records upon running the long running programs more than once. If the records from these tables (ABSADDR and NSACL) are not deleted then run the db_convert_nsac1 and db_convert_vvtape conversions prior to running the corresponding long running program again. The programs that perform load inserts are:

- db_convert_nsubject
- db_convert_storagesegtape

The db_convert_nsubject program performs a load insert into the DB2 subsystem table NSACL and depends on db_convert_nsac1 (part of db_subsys_convert) to be run first. The db_convert_storagesegtape program performs a load insert into the DB2 subsystem table ABSADDR and depends on db_convert_vvtape (part of db_subsys_convert) to be run first.

To determine whether the programs completed successfully, search the output file for keywords such as "Error:" and "Warning:". See Section 6.2.7: Capturing the Metadata Conversion Output on page 185 and Section 6.4: *Metadata Conversion Troubleshooting Procedures* on page 227. If db_config_convert, db_subsys_convert and all individual long running programs complete successfully, the metadata conversion is finished.

Restarting the Long Running Utilities

The only conversion programs with restart capability are the long running conversion programs. To use the restart capability, run the program with a "-restart" flag. The command to restart will look like this:

```
db_convert_bitfile -g /./encina/sfs/hpss/globalconfig \
```

```
-db /var/hpss/convert/6.2/Convert.DB.Names \  
-sub 1 -restart
```

When given the '-restart' flag, the program will first attempt to determine that no other conversions or DB2 loads are currently running on the specific table. If a load or conversion is still running (due to failure to cleanup from an abnormal termination), the conversion should output a message and quit. If no conversion or load is running, the restart will perform a db2 load insert to append metadata to the table. The output will be slightly confusing in that it will show that it has submitted a total number of records to the load that will differ from the db2 load output of how many records were read, loaded and committed to the table. The difference should be exactly the number of records present in the table before executing a restart. For example, suppose a long running conversion program is terminated with a CTRL-C. The output might look like the following:

```
Running lr_db_convert_storagesegdisk utility...  
                               Converting SFS file  
././encina/sfs/testsf/s/storagesegdisk..^C  
Error: User termination (CTRL-C).
```

```
Number of rows read           = 22000  
Number of rows skipped        = 0  
Number of rows loaded          = 22000  
Number of rows rejected        = 0  
Number of rows deleted         = 0  
Number of rows committed      = 22000
```

This output indicates that the program terminated from a CTRL-C, but the db2 load already placed 22000 storagesegdisk records in the storagesegdisk table (Number of rows committed is the only significant number for determining how many records are actually in the table). The program is ready to be restarted:

```
Re-running lr_db_convert_storagesegdisk utility...  
Converting SFS file  
././encina/sfs/testsf/s/storagesegdisk.....  
.....  
                               Converted 376923 records successfully from  
././encina/sfs/testsf/s/storagesegdisk  
lr_db_convert_storagesegdisk complete, submitted 376923 records to  
DB2 storagesegdisk load for subsystem 1, 1389 operations per sec,  
271.308885 total time
```

```
Number of rows read           = 354923  
Number of rows skipped        = 0  
Number of rows loaded          = 354923  
Number of rows rejected        = 0  
Number of rows deleted         = 0  
Number of rows committed      = 354923
```

Notice that the 'Number of rows committed' upon restarting is only 354923, which is 22000 less than the total number of records in the SFS file, 376923. That is because the restarted load doesn't know that 22000 records were already loaded into the table. This was a successful restart.

The restart option works very well if an expected error occurs, such as running out of disk space or an

error occurs while reading metadata. A restart from a user termination (CTRL-C) works well 90% of the time. Sometimes a user termination can cause DB2 to output many system log errors (usually means the db2as, administrative server died). If the program is unable to perform a restart (i.e. an attempt to perform the restart and the program hangs or exits with an error again), there are steps to take to discover the problem.

First, see if there is a load pending on the table:

```
$ su hpss
$ db2
db2> connect to <Database Name>
db2> list tablespaces
```

Scan the output for the table space that corresponds to the table where the failed conversion was performed. Under the table space information, find Detailed explanation. If the Detailed explanation is Normal, perform a restart at this time because there is no load pending. If the Detailed explanation is Load Pending or Quiesced, there is a load pending on the table. Determine what state the load is in. Do this by performing a load query on the table to be restarted:

```
$ su hpss
$ db2
db2> connect to <Database Name>
db2> load query table <Table Name>
```

The load query result could be one sentence or several screens worth of SQL messages. Examine the output and try to determine what phase the load is in by finding the last SQL3500 message in the output. At this point, the load will either be in the REBUILD or LOAD phase. The load phase provides information to help understand how to reset the table, if the restart fails and the restart attempt is abandoned. At this point, attempt to perform the restart. The restart on a load pending can take a long time depending on how many records have already been inserted into the table. To check the status of the load, execute a load query command:

```
$ su hpss
$ db2
db2> connect to <Database Name>
db2> load query table <Table Name>
```

Run the load query several times in a row to determine if the load is counting records properly. Look at the number next to Number of rows read, it will count up until it reaches the Number of rows loaded and only then will the conversion program continue with normal processing from the last record in the DB2 table. If there is no progress, the program is hung; terminate it and abandon restart procedures.

After abandoning restart procedures, attempt to run the conversion program without the '-restart' option. If the program fails, try terminating a pending load. To terminate a load, run the program 'load_terminate_*' where * is the name of the table that was being converted. This should terminate any pending load operations on that table. If after rerunning the program and there are still problems, consider restarting DB2 or rebooting the system.

6.3.13.6. Create Core Server ACLs

Prior to running this conversion program. Ensure you temporarily set the following



environment variables to HPSS 4.5 values. Examples follow:

- `HPSS_PRINCIPAL_SSM=hpss_ssm`
- `HPSS_PRINCIPAL_FTPD=hpss_ftp`
- `HPSS_PRINCIPAL_MPS=hpss_mps`
- `HPSS_PRINCIPAL_NDCG=hpss_ndcg`
- `HPSS_PRINCIPAL_NFSD=hpss_nfs`
- `HPSS_PRINCIPAL_DMG=hpss_dmg`

This allows the `db_convert_dce_cds` program to add proper DCE principals from HPSS 4.5 to a newly created core server ACL. Upon completion of the `db_convert_dce_cds` program, ensure the values are returned to valid HPSS 6.2 values.

Run the `db_convert_dce_cds` script.

```
% db_convert_dce_cds -sub <subsys id> \  
    -g /.: /encina/sfs/hpss/globalconfig
```

The `db_convert_dce_cds` utility adds the DCE principal and account and sets the CDS Security ACLs for the new 6.2 Core Servers. It also updates the CDS Security ACLs for the other HPSS servers to allow access for the Core Servers. `db_convert_dce_cds` creates the Core Server security ACLs with the permissions listed below, and adds the Core Server principal to the Gatekeeper, DMG, and PVL servers. The utility does not modify or remove any other HPSS server security ACL settings. This will allow the existing HPSS 4.5 system (even one with incorrect security settings) to run after the system has been upgraded to HPSS 6.2 just in case the upgrade needs to be reversed.

- `hpss_core` security ACL:
 - {user cell_admin:rwdtc}
 - {user hpss_ftp:rc}
 - {user hpss_ndcg:rc}
 - {user hpss_dmg:rwc}
 - {user hpss_ssm:rwc}
 - {user hpss_mps:rwdtc}
 - {user hpss_nfs:rc}
 - {user any_other:t}

6.3.13.7. Terminate the Scripting Session

Terminate the scripting session.

```
% exit
```

6.3.13.8. Modify Permissions on Devices

It may be necessary to modify permissions on the devices configured on the HPSS server



machine to enable the HPSS 6.2 system to work properly. On several test machines, we had to open world access to the devices used by the HPSS Core Server and PVL in order for them to initialize properly.

6.3.14. Verify HPSS 4.5 Conversion Results



- Prior to running the verification utilities, we recommend that running the DB2 utility `RUNSTATS` on each table in every database. This will enable the verification utilities to run quicker than if `RUNSTATS` is not used. For example: “`db2 runstats on table hpss.bitfile and indexes all allow write access`”.
- Prior to running the verification utilities, set the environment variables `HPSS_PRINCIPAL_SSM` and `HPSS_KEYTAB_FILE_SERVER` back to their HPSS 4.5 values as the verification programs need to authenticate with SFS with the proper user credentials. For example, at most sites the `HPSS_PRINCIPAL_SSM` should be `hpss_ssm` and the `HPSS_KEYTAB_FILE_SERVER` should be `/krb5/hpss.keytabs`. After completing the verifications, these environment variables should be returned to their HPSS 6.2 values.

There are three utilities provided to verify the correctness of the HPSS 4.5 to 6.2 converted metadata. They are located under `$HPSS_ROOT/convert62/bin`.

- `db_convert_size_check`
- `db_convert_ns_check`
- `db_convert_address_check`

Different degrees of verification may be performed based on the amount of time an HPSS site decides to devote to this task. Verification is highly recommended immediately following the conversion before the 6.2 HPSS servers are started.

Simply executing the verification programs will run them with the recommended or default option. See the utility's usage for additional options which are available. Perform the following steps for each root and non-root subsystem:

6.3.14.1. Capture Session Output

Set up a scripting session to capture the verification output.

```
% script verify.output
```

6.3.14.2. Run `db_convert_size_check`

Run the `db_convert_size_check` script.

```
% db_convert_size_check -g ././encina/sfs/hpss/globalconfig
```

The utility `db_convert_size_check` is provided to verify the existence and size of all the expected DB2 tables following the conversion. This utility checks that all of the expected tables exist and checks the number of records in each table against the number of records in the corresponding SFS metadata file(s). This utility runs quickly and should be run at every site. The default behavior of this utility is to check all metadata tables. If the “-t” option is specified, the utility will only check the given table. If the “-m” option is specified, the utility will perform a metadata consistency check to

ensure that the site has SFS and DB2 configured as expected. This program is not able to check the Name Server Objects (NSOBJECT) or ACLs (NSACL) table. The object count between two systems does not match because objects associated with deleted bitfiles are not converted. To invoke this utility, see the usage below:

```
db_convert_size_check
  -g <SFS Global Config File>
  -db <DB Names Text File>
  [-m] (Metadata Consistency Check - sfs vs. db2)
  [-t <Table Name>] (check specified table)
  [-v] (verbose)
```

6.3.14.3. Run db_convert_ns_check

Run the db_convert_ns_check script.

```
% db_convert_ns_check -g ./:/encina/sfs/hpss/globalconfig
```

The db_convert_ns_check utility is provided to perform a field level check of the SFS Name Server metadata against the new Core Server metadata. If all tables are checked, the amount of time required is expected to be slightly less than that of the conversion itself. The default behavior of this utility is to check the first and last 1000 nsubject records. It is expected that all sites should run this utility with at least the default behavior. If the SFS system that was converted was in read-only mode, certain timestamps and other metadata may not match between the two systems. The -a flag directs this program to check all nsubject records in the DB2 table. The -f and -l flags followed by a number will direct the program to check the first and last number of records specified in the nsubject SFS file against the matching records in DB2. The -v option serves as both a verbose and force option to this program. With the -v option, the program will output errors but continue executing until normal program termination. It will then provide an error summary, reporting the type and number of errors encountered. To invoke this utility, see the usage below:

```
db_convert_ns_check
  -g <SFS Global Config File>
  -db <DB Names Text File>
  -s <Subsystem Id>
  [-a] (check all nsubject records)
  [-f <Number>] (check first <Number> records)
  [-l <Number>] (check last <Number> records)
  [-v] (verbose and continue through errors)
```

6.3.14.4. Run db_convert_address_check

Run the db_convert_address_check script.

```
% db_convert_address_check -g ./:/encina/sfs/hpss/globalconfig
```

db_convert_address_check is provided to take HPSS bitfile descriptors and resolve them to media volume names and addresses at each level in the hierarchy in which the file resides. The utility determines these addresses using both SFS and DB2 metadata and compares the results. Options are available to check a subset of files in the system, such as the first and last 1000 files, or every file in the system. Running a full check using this utility takes slightly less time than the conversion itself. It is expected that all sites should check a subset of their user files using this utility. The default behavior of this utility is to check the first and last 1000 files. The -a option forces this program to

check every bitfile descriptor in the 6.2 system. The -f and -l options check only the first and last number of bitfile descriptors provided with these options. To invoke this utility, see the usage below:

```
db_convert_address_check
  -g <SFS Global Config File>
  -db <DB Names Text File>
  -ss <SSId Text File>
  -s <Subsystem Id>
  [-a] (check all files)
  [-f <Number>] (check first <Number> files)
  [-l <Number>] (check last <Number> files)
```

6.3.14.5. Terminate Scripting Session

Terminate the scripting session. This ends the process for converting HPSS 4.5 specific metadata. Proceed to Section 6.3.17: *Perform the DCE Export: hpss_dce_export* on page 215 to continue the conversion to HPSS 6.2.

```
% exit
```

6.3.15. Upgrade from HPSS 5.1 to HPSS 6.2

Only follow procedures in this section if upgrading a system from HPSS 5.1 to HPSS 6.2.

To upgrade HPSS from version 5.1, run the `hpss_md_convert_51` utility. The `hpss_md_convert_51` program is a menu driven program that converts the 5.1 HPSS metadata to HPSS 6.2 metadata. It does not modify HPSS 5.1 metadata so that it is possible to revert to the HPSS 5.1 system during any point in the conversion process. The metadata changes are performed by renaming the 5.1 HPSS table by pre-pending the original table name with "PRE62_" and creating a new HPSS 6.2 table. The conversion then reads information from the HPSS 5.1 tables, makes the necessary metadata modifications in memory, and inserts the new metadata into the HPSS 6.2 table. In cases where a function is no longer supported, the pre-6.2 table is renamed pre-pending the original table name with "PRE62_", but the new table in HPSS 6.2 is not populated with metadata. All steps must be followed in order to successfully convert the metadata from HPSS 5.1 to HPSS 6.2.

To begin the metadata conversion, run the conversion program `hpss_md_convert_51`:

```
% hpss_md_convert_51
HPSS 5.1 - 6.2 Metadata Conversion
1) Rename old tables
2) Create new tables
3) Convert Gatekeeper config table
4) Convert Location Server policy table
5) Convert Server Generic config table
6) Convert DMG config table
7) Convert Mover Device table
enter number or RET to exit:
```

If errors occur, see Section 6.4: *Metadata Conversion Troubleshooting Procedures* on page 227 of this document before contacting your HPSS customer support representative.

Successful output for each option follows:

Option 1 – Rename old tables

```
[ renaming LSPOLICY table to PRE62_LSPOLICY ]  
[ renaming SERVER table to PRE62_SERVER ]  
[ renaming DMG table to PRE62_DMG ]  
[ renaming GATEKEEPER table to PRE62_GATEKEEPER ]  
[ renaming MOVERDEVICE table to PRE62_MOVERDEVICE ]  
[ renaming SITE table to PRE62_SITE ]
```

Option 2 – Create new tables

Note: This option uses the `hpss_managetables` program which will assist in creating the newly formatted HPSS 6.2 tables.

Execute the following commands to complete this step:

```
db cfg  
add LSPOLICY  
add SERVER  
add DMG  
add GATEKEEPER  
add MOVERDEVICE  
add SITE  
add SERVERINTERFACES  
add AUTHZACL  
db subsys1  
add DMGFILESET  
commit  
quit
```

Option 3 – Convert Gatekeeper config table

```
[ converting PRE62_GATEKEEPER to GATEKEEPER ]
```

Option 4 – Convert Location Server policy table

```
[ converting PRE62_LSPOLICY to LSPOLICY ]
```

Option 5 – Convert Server Generic config table

```
[ converting PRE62_SERVER to SERVER ]
```

Option 6 – Convert DMG config table

```
[ converting PRE62_DMG to DMG ]
```

Option 7 – Convert Mover Device table

```
[ converting PRE62_MOVERDEVICE to MOVERDEVICE ]
```

This ends the process for converting HPSS 5.1 specific metadata. Proceed with Section 6.3.17: *Perform the DCE Export: `hpss_dce_export`* on page 215 to continue with the conversion to HPSS 6.2.

6.3.16. Enable DB2 Backup

Set up the DB2 backup process for HPSS metadata. Ensure that an initial backup of all HPSS databases are available at this point. Perform this step on each root and non-root subsystem.

6.3.17. Perform the DCE Export: `hpss_dce_export`

This step applies to upgrades for both HPSS 4.5 and 5.1. The program that exports principal and group information from DCE is called `hpss_dce_export`. The program sends its output to ASCII text files located in the output directory designated as an option. Create a subdirectory called `/var/hpss/convert/6.2`, if one doesn't already exist. Using the new subdirectory, invoke the DCE export program as follows:

```
Usage: hpss_dce_export <output dir>
```

```
% hpss_dce_export /var/hpss/convert/6.2
```

The program will start by looking for the existence of a file, called `skipFile`, in the path provided (e.g. `/var/hpss/convert/6.2`). This file contains information on principals and groups in DCE that should or should not be exported from DCE. If the file doesn't exist, the first execution of this program will create a default `skipFile`.



- *If the `skipFile` does not exist during the first execution, the `hpss_dce_export` script will need to be re-executed in order to export data from DCE.*
- *Each site is highly recommended to use the default `skipFile` only after close analysis. Many principals and groups in DCE are not necessary for HPSS 6.2 to run properly; therefore, accepting the default values in the `skipFile` will probably result in the addition of many unnecessary principal and group entries to the LDAP or Unix registry*

When the `skipFile` exists or when the program is executed the second time, the program will export principal, group, and cell information from DCE into the output directory provided. These files will be read as input to the import program corresponding to the authorization mechanism each site will utilize.

6.3.18. Perform the Unix, LDAP or Kerberos Import

This step applies to conversions for both HPSS 4.5 and 5.1. See the following sections on information on how to import the DCE authorization information into Unix, LDAP or Kerberos. The necessary steps to be performed depend on the authorization mechanism selected by the site. Note that the `hpss_ldap_admin` utility must be run after the `hpss_ldap_import` utility.

Import DCE Authorization Information into Unix for HPSS Servers

The conversion program that imports HPSS server information into the Unix environment is called `hpss_unix_servers`. The `hpss_unix_servers` program supports Unix authorization by gathering information for each HPSS server from DCE and importing it into the Unix or HPSS password, group and shadow files. The `hpss_unix_servers` program also creates a Unix keytab file (usually in `/var/hpss/etc/hpss.mm.keytab`) and adds the HPSS server principals to the keytab file for initial startup. Passwords for HPSS servers in the keytab file are set initially to default values then encrypted.

There are no command line options for the `hpss_unix_servers` program. Before running the program, ensure that the environment variables for HPSS server principals (e.g. `HPSS_PRINCIPAL_CORE`) are set to proper HPSS 6.2 values. If non-default server principal values were used in HPSS 4.5 or

5.1, set the non-default server principal values using the appropriate environment variable names:

- HPSS_PRE62_PRINCIPAL_CORE
- HPSS_PRE62_PRINCIPAL_DMG
- HPSS_PRE62_PRINCIPAL_FTPD
- HPSS_PRE62_PRINCIPAL_GK
- HPSS_PRE62_PRINCIPAL_HPSSD
- HPSS_PRE62_PRINCIPAL_LOG
- HPSS_PRE62_PRINCIPAL_LS
- HPSS_PRE62_PRINCIPAL_MOUNTD
- HPSS_PRE62_PRINCIPAL_MPS
- HPSS_PRE62_PRINCIPAL_MVR
- HPSS_PRE62_PRINCIPAL_PVL
- HPSS_PRE62_PRINCIPAL_PVR
- HPSS_PRE62_PRINCIPAL_SSM
- HPSS_PRE62_PRINCIPAL_NFSD
- HPSS_PRE62_GRP_NAME_SERVER

Run the `hpss_unix_servers` program:

```
% hpss_unix_servers
```

Import DCE Authorization Information into Unix for HPSS Users

The conversion program that imports HPSS user information into the Unix environment is called `hpss_unix_import`. The `hpss_unix_import` program supports the Unix authorization by gathering authorization information (UID, moniker, and home directory) for each HPSS user from DCE and importing it into the Unix or HPSS password, group and shadow files. The `hpss_unix_import` program also optionally supports Unix authentication by obtaining the HPSS user passwords from DCE and imports them into the Unix or HPSS password file.

To run the utility, specify the path name to the location of the previously exported DCE information as follows:

```
Usage: hpss_unix_import <input dir>
```

```
% hpss_unix_import /var/hpss/convert/6.2
```

where `<input dir>` is the directory name specified previously as the `<output directory>` for the `hpss_dce_export` program. See Section 6.3.17: *Perform the DCE Export: `hpss_dce_export`* on page 215.

Import DCE Authorization Information into LDAP using `hpss_ldap_import`

The conversion program that imports DCE authorization information into LDAP is called `hpss_ldap_import`.

```
Usage: hpss_ldap_import <input dir> -realmname <realm>
```

```
% hpss_ldap_import /var/hpss/convert/6.2 -realmname  
"cn=hpss.acme.com"
```

Where the `realmname` option should use the name of the realm desired in LDAP.

The program requires a path to the directory where expected input files reside (the same path used when running `hpss_dce_export`). The program also allows some options for specifying what should be imported. Executing the program with no optional commands will result in a full import of group, principal, and cell information into LDAP. Sites are only recommended to use options if previous steps fail and only part of the import should take place.

The program may output warnings like "WARNING: this group has no members". The groups are still properly imported exactly as they existed in DCE (i.e. with no members), but the warning may help the site determine if the group really is necessary or not in HPSS 6.2.

The `hpss_ldap_admin` utility must be run following the `hpss_ldap_import` utility.

Create Local Site Information using `hpss_ldap_admin`

There is no utility provided to convert the local site information from the Location Server Policy (LS Policy) into LDAP. However, the Location Server needs to be able to lookup the local site entry in LDAP to register endpoints with the RPC group to successfully initialize and start in HPSS 6.2. Use the new LDAP administration tool, `hpss_ldap_admin`, to create a new site entry using the correct local site name from the Location Server Policy in HPSS 4.5 or 5.1. For example, if the local site name was "hpss.acme.com":

```
% hpss_ldap_admin  
LDAP: connected to hpss.acme.com:389  
realm: cn=hpss.acme.com  
hla> site create -name hpss.acme.com  
dn: cn=hpss.acme.com,cn=hpssSite,cn=hpss.acme.com  
return code: 0 (HPSS_E_NOERROR)
```

In the example above, the `hpss_ldap_admin` program created a new site entry called "hpss.acme.com" to match the local site name in HPSS 4.5 or 5.1 from the LS Policy metadata.

Import DCE Information into Kerberos

There is not utility provided to convert DCE principals and their passwords and UIDs into Kerberos. Instead, sites should consider creating new Kerberos accounts for each DCE principal that requires access to HPSS 6.2 that will have new Kerberos passwords. A site could create a Kerberos keytab file in the event that users aren't required to know a password to access HPSS.

6.3.19. Prepare the 6.2 System



The following steps should be performed only if the metadata conversion completed successfully and no errors were reported by the conversion verification utilities. If there is a possibility that the HPSS system will be reverted back to the 4.5 or 5.1 level, do not attempt to continue with the remaining conversion steps.

6.3.19.1. Tune DB2 for normal operations

Refer to Section 5.7: *Tune DB2* on page 171. Perform this step on each root and non-root subsystem.

6.3.19.2. Modify Accounting, if applicable

If the HPSS system consists of multiple subsystems and accounting by subsystems is desired, accounting metadata should be modified to support accounting by subsystem. The necessary procedure is quite complex. Contact your HPSS customer support representative for information. Perform this step on each root and non-root subsystem.

6.3.19.3. Update FTP Configuration Files

Perform this step on the root subsystem only if PFTP Daemon is to run there.

1. Save the `/etc/inetd.conf` file, where `conf_xx = conf_45 or conf_51`.

```
% cd /etc  
% cp -p inetd.conf inetd.conf_xx
```
2. Remove the obsolete flags “-LSUXddd” from the “hpsftp” entry in the `/etc/inetd.conf` file.
3. Refresh the `inetd` Daemon.

```
% refresh -s inetd, or  
% kill -1 <inetd pid>
```

6.3.19.4. Populate the HPSS.conf files

The `mkhps` utility should have created the `/var/hpss/etc/HPSS.conf` file. Transfer any customized data from the 4.5 or 5.1 `HPSS.conf` file to the `HPSS.conf` file for the HPSS 6.2 system (`/var/hpss/etc/HPSS.conf`). Perform this step on the root subsystem only and then place the `HPSS.conf` file on other platforms with appropriate changes.

6.3.19.5. Copy the rc.hpss Script to /etc

Set up the `rc.hpss` script in `/etc` (perform on each HPSS machine).

```
% mv /etc/rc.hpss /etc/rc.hpss.pre62  
% cp -p /opt/hpss/bin/rc.hpss /etc
```

6.3.19.6. Run the bind Script

Run the `bind` script or use the `mkhps` utility to perform the `bind`. The `bind` operation allows fileset operations to succeed in HPSS. Perform this step on the root subsystem only.

```
% /opt/hpss/bin/hpss_db2_bindall.ksh
```

6.3.19.7. Create Default Server Security ACLs

Perform this step on the root subsystem only. Upon performing a successful metadata conversion from HPSS 4.5 or 5.1 to HPSS 6.2, each site will need some modifications of their ACL settings in order to allow each server to initialize and run. To initialize the ACLs for HPSS 6.2 servers, run the `hpss_init_server_acls` program located in `HPSS_ROOT/bin/convert62`. This creates the default server security ACLs based on the default server interfaces. For each HPSS server, there is a single or a set of server interfaces, a set of authorized callers for each interface, and a default set of permissions

that each caller of the interface is given.

Run the `hpss_init_server_acls` program as follows:

```
% /opt/hpss/bin/convert62/hpss_init_server_acls
Error in stat of Keytab File, /var/hpss/etc/mm.keytab, 2
```

Note: The error is expected and does not indicate that the `hpss_init_server_acls` program did not complete successfully. In the above example, the error is displayed because the system did not have a `mm lib` or `DB2` username and password setup for this system.



We recommend checking each server's ACLs at this time and ensuring that you fully understand the reasons for granting any group ACLs on any server. See Section 2.1: HPSS Server Security ACLs of the HPSS Management Guide for further information.



The `hpss_init_server_acls` program will destroy any existing entries in the `AUTHZACL` table. Therefore, this step should be executed before manually adding SSM users to the `AUTHACL` table, which is explained in section 6.3.19.8: Create SSM User Ids.

6.3.19.8. Create SSM User Ids

Create new SSM user Ids. There is no utility provided to convert the SSM user Ids from HPSS 4.5 or 5.1. Perform this step on the root subsystem only.

SSM User Ids under HPSS 4.5:

As of HPSS 5.1, SSM only supports the “admin” and “operator” security levels. If a 4.5 SSM ID is assigned an obsolete security level (“user” or “privileged”), then it must be changed to either “admin” or “operator”. See `/var/hpss_45/ssm/hpssadm.config` and `/opt/hpss_45/sammi/hpss_ssm/user_authorization.dat` files for a list of valid 4.5 SSM users.

SSM User Ids under HPSS 5.1:

See `/var/hpss_51/ssm/ssmuser.config` for a list of valid HPSS 5.1 SSM users.

Run `hpssuser` Utility

For each user, client platform, and security mechanism combination, run `hpssuser` utility and provide the necessary information. For an example on adding SSM user ‘fred’, see below:

```
% /opt/hpss/bin/hpssuser -add fred -ssm
[ adding ssm user ]
1) admin
2) operator
Choose SSM security level
(type a number or RETURN to cancel):
> 1
[ ssm user added : admin ]
```

Once ssm users are created, a ssm client package can be created. For example:

```
% /opt/hpss/bin/hpssuser -ssmclientpkg /tmp/ssmclientpkg.tar
[ packaging ssm client ]
[ creating /tmp/ssmclientpkg.tar ]
ssm.conf
```

```
hpssgui.pl
hpssgui.vbs
hpss.jar
login.conf
```

The `-ssmclientpkg` option creates a tar file and places it in the specified directory (e.g. `/tmp/ssmclientpkg.tar`) which will contain the following files:

- `hpss.jar` – Java executables for running the SSM GUI
- `hpssgui.pl` – Perl version of the SSM GUI start up script
- `hpssgui.vbs` – Visual Basic version of the SSM GUI start up script
- `krb5.conf` – Kerberos environment variable file
- `login.conf` – SSM user login configuration file for HPSS users on AIX platform
- `ssm.conf` – Contains environment variables required for the user to contact SSM using Unix authentication

These files will need to be installed on each SSM client. See Section 3.3.5.2: *Manual SSM Client Packaging and Installation* of the Management Guide for more information. The program also creates SSM user ACLs for the SSM server in the new AUTHZACL table granting permissions to each SSM user based on their level (admin or operator).

6.3.19.9. Create Location Server Endpoints

The endpoint utility, `hpss_bld_ep`, creates endpoints and places the information in the file `/var/hpss/etc/ep.conf` for the Location Server to use in HPSS 6.2. The utility is located in `$HPSS_ROOT/config`. Perform this step on the root subsystem only.

To invoke the utility:

```
% /opt/hpss/config/hpss_bld_ep -v -f /var/hpss/etc/ep.conf -r <Unix/
LDAP Realm Name> -c <schema name> -d <DB2 configuration database>
add
```

For example,

```
% /opt/hpss/config/hpss_bld_ep -v -f /var/hpss/etc/ep.conf -r
hpss.acme.com -c hpss -d cfg add
```

6.3.19.10. Perform Additional Remote Mover Configuration

Perform the required additional configuration on each remote mover node. See the HPSS Management Guide, Section 5.2.8.2: *Additional Mover Configuration*.

6.3.20. Bring up the HPSS 6.2 Servers



At this point, verify that all conversions steps were performed successfully on each subsystem before bringing up the 6.2 servers.

The next step is to bring up the HPSS Servers. Before all the servers can be started, some configuration settings need to be review and/or modified and the accounting metadata should be dumped.

6.3.20.1. Invoke the SSM System Manager, Startup Daemon and prerequisite software

Perform this step on the root subsystem only. Use the rc.hpss script to invoke the Startup Daemon, SSM System Manager and the prerequisite software.

```
% /opt/hpss/bin/rc.hpss
```

or

```
$ /etc/rc.hpss
```

If problems are encountered, try starting the software independently using the rc.hpss script. The default operation is "start".

```
Usage: rc.hpss [-o] [-m] [-d] [-p] [start | stop]
        -o - direct output to /dev/console
        -m - do only System Manager
        -d - do only Startup Daemon
        -p - do only prerequisite software
```

Additionally, ensure that the environment variables HPSS_PRIMARY_AUTHN_MECH and HPSS_PRIMARY_AUTHENTICATOR are defined in /var/hpss/etc/env.conf and an authentication mechanism is defined in /var/hpss/etc/site.conf. See Section 6.3.10: *Setup Authentication and Authorization* on page 199 for more information.

The SSM System Manager obtains the environment variable values from various sources. It will attempt to read the variables, if set, in the following order:

- Variables which are set in the environment of the shell from which rc.hpss is executed
- /var/hpss/etc/env.conf
- Default setting (\$HPSS_ROOT/include/hpss_env_defs.h)

If the System Manager will not start, check the value of the parameters which are displayed in the command line output.

Start an SSM Client Session

Perform this step on the root subsystem only. Start an SSM Client session on the main HPSS server:

```
% su - <userid>
% /opt/hpss/bin/hpssgui.pl
```

If problems are encountered, verify the values of the following environment variables:

```
HPSS_SSM_SEC_MECH
HPSS_SSM_RPC_PROT_LEVEL
HPSS_SSM_UNIX_REALM
HPSS_SSM_SM_HOST_NAME
HPSS_SSM_SM_PORT_NUM
```

These values are typically set in the SSM configuration file. The values may be overridden by hpssgui.pl command line parameters or by setting the variables in the user's environment. If a value is not specified by one of these means, it is taken from /var/hpss/etc/env.conf or, if not there, from the default in \$HPSS_ROOT/include/hpss_env_defs.h.

The default name of the SSM configuration file is `ssm.conf`. Its default location is the current directory or, if not there, in `$HPSS_PATH_SSM` (`/var/hpss/ssm` by default). The name and location may be overridden by the `-m` option to the `hpssgui.pl` script or by setting the value in the user's environment.

The value of `HPSS_SSM_SM_PORT_NUM` can be obtained by connecting to the DB2 configuration database (e.g. CFG) and issuing the following SQL statement:

```
% db2 select rpc_prog_num, rpc_prog_vers from hpss.server where
"desc_name='SSM System Manager'"
```

```
RPC_PROG_NUM  RPC_PROG_VERS
-----
536870918      1
```

```
1 record(s) selected.
```

In this example, the `HPSS_SSM_SM_PORT_NUM` should be set to the value `"536870918:1"` for the SSM client to successfully connect with the SSM server. The `rpcinfo` command can be used to verify that a server is listening on this port.

6.3.20.2. Update HPSS Configurations

Perform this step on the root subsystem only.

Using the SSM GUI,

1. Mark the following servers non-executable:
 - NFS Daemon (not yet supported for this release)
 - Mount Daemon (not yet supported for this release)
 - MPS (to temporarily disable migration and purge)
2. Delete DMAP Gateway server, if configured
3. Review each HPSS server configuration and update them to match with the 6.2 default configuration values shown in the following table:

Table 11. 6.2 Default Server Configuration Parameters

Server Type	Thread Pool Size	Maximum Connections	Minimum Database Connections	Maximum Database Connections
Core Server	200	200	10	50
Location Server	200	200	3	10
Log Daemon	10	10	0	2
Log Client	10	10	0	2
Gatekeeper	200	200	1	5

MPS	10	10	5	20
PVL	50	10	1	2
PVR	30	20	5	30
Mover	20	20	1	5
Startup Daemon	10	10	0	2
SSM System Manager	100	100	5	20

4. Review and update the Core Server(s) other configuration parameters, if necessary
5. For 4.5 upgrades only: Review all migration policies to ensure that they were converted correctly

In pre-5.1 HPSS systems, migration policies could apply to both a disk and tape storage class. In HPSS 5.1, the migration policies were modified to apply only to disk or to tape, but not both. The conversion utility examines each migration policy and attempts to determine whether the policy was intended to be used by a disk or a tape storage class. If the policy is in use by both a disk and tape storage class, the program will produce a new but identical migration policy and classify one of the policies as type disk and the other as type tape, then update the appropriate storage class to use the correct migration policy. Benign warnings from the `db_convert_mps` utility may be observed indicating that a given migration policy id has both disk and tape flags set. In this case, the program will leave both flags set but output this warning. To correct, the incorrect flag should be unset using SSM. Additionally, the program may also warn when a migration policy is not in use by any storage class. In this case, the program will designate this migration policy as disk and leave it unassigned.

6.3.20.3. Dump Accounting Metadata, if applicable

At this point, the accounting metadata has been converted. For a single-subsystem HPSS, that conversion is sufficient. For a multiple-subsystem HPSS, perform the following steps:

1. For each subsystem database, empty the `acctsum`, `acctlog`, and `acctsnap` tables. This is accomplished by using the following commands:

```
db2 connect to <db> (e.g., "db2 connect to subsys1")
db2 set schema hpss
db2 delete from acctsum
db2 delete from acctlog
db2 delete from acctsnap
```

Note that this is done once for each subsystem and that `<db>` is replaced by the name of the subsystem database.

2. Swap the accounting bits for each Core Server.

There are two accounting bits for each bitfile. One bit is active at a time; the Core Server's specific configuration indicates which bit is currently active.

For each Core Server, use the `load_core_config` utility located in the `tools/load` directory and update the specific configuration to use the other bit (the one not currently being used). This is done by invoking `load_core_config` and selecting the proper subsystem. The configuration

fields will display the current value. Change the accounting bits such that if the initial value is 1 then the new value will be 2 and conversely if the initial value is 2 then the new value will be 1.

At this point, the HPSS system has no accounting metadata for existing files. Any new files that are added will generate new accounting metadata but existing files need to have their accounting metadata created using an additional conversion routine (`acct_convert`).

Since the final step of the accounting conversion is time-consuming, we recommend that it be postponed until a later time and the *Start HPSS 6.2 Servers* step, Section 6.3.20.4: below, be performed next. Verify that HPSS functions properly and that the conversion has been successful before performing the third and final step in converting the accounting metadata:

3. After it is clear that the system will not be reverted, use the `/opt/hpss/bin/acct_convert` utility to regenerate the accounting metadata for each subsystem's existing files. This utility must be run once for each subsystem and will run for some time depending upon the size of the HPSS system (possibly for multiple days). If a restart is necessary, built-in checkpoint capabilities will allow it to resume processing. Be sure to run the utility to completion, and check the utility's output to ensure that it has completely converted all the necessary subsystem accounting metadata

6.3.20.4. Start HPSS 6.2 Servers

If using unix authentication, all HPSS servers must run as root. If necessary, modify each server's configuration's Username to root using the SSM Server Configuration window.

Using the SSM client, start up all the configured HPSS 6.2 servers. After starting other HPSS servers, error messages shown on the console include:

```
Error in stat of Keytab File, /var/hpss/etc/mm.keytab, 2
```

To resolve this problem, create an empty file owned by root with 644 permissions.

6.3.21. Verify 6.2 System



If there is a possibility that the HPSS system will be reverted back to the 4.5 or 5.1 level, do not attempt to continue with the remaining upgrade steps. Doing so may result in corruption of the 4.5 or 5.1 user data

Perform the following steps for each root and non-root subsystem:

- Read existing files using each configured user interfaces
- Write new files to HPSS using each configured user interfaces
- Restore all execution permissions for the following binaries in the `/opt/hpss/bin` directory:
 - `hpss_mps`
 - `repack`
 - `reclaim`
 - `recover`
 - `shelf_tape`
 - `shelf_tape_util`

- Using the GUI, mark the MPS configuration executable and start it
- Verify that Migration and Purge are working correctly
- Repack and reclaim tape volumes

6.3.22. Clean Up After a 4.5 to 6.2 Upgrade

After the 6.2 system has been operational for an extended period, clean up the following:

- If it is still running, shutdown the SFS server
- Unconfigure SFS server(s) using mkhpss from the /opt/hpss_45 code
- Uninstall Encina code
- Uninstall Sammi code
- Delete HPSS 4.5 code saved in /opt/hpss_45
- Delete the /var/hpss_45 directory
- Delete all files and subdirectories in the /var/hpss/convert/6.2 directory. Do not delete this information until all errors or warnings have been resolved
- Run the db_convert_dce_cleanup utility to eliminate obsolete 4.5 DCE entries. Only run the utility if it is certain that the HPSS 4.5 system will never be used again on the upgraded machine

6.3.23. Clean Up After a 5.1 to 6.2 Upgrade

After the 6.2 system has been operational for an extended period, clean up the following:

- Delete HPSS 5.1 code (e.g. /opt/hpss_51)
- Delete the /var/hpss directory for HPSS 5.1 (e.g. /var/hpss_51)
- Delete all files and subdirectories in the /var/hpss/convert/6.2 directory. Do not delete this information until all errors or warnings have been resolved

6.3.24. Revert HPSS 6.2 System to Prior Release

In certain situations, a decision may be made to abort the upgrade and revert the HPSS system to the 4.5 or 5.1 level. This decision can only be made after carefully considering the sequence of the upgrade steps. Failure to do so may result in corruption of the HPSS 4.5 or 5.1 user data.



If a reversion is deemed necessary after the 6.2 system is up and running, it is strongly recommended your HPSS customer support representative be consulted.

6.3.24.1. Revert the HPSS 6.2 System to Version 4.5

The 6.2 conversion utilities do not alter the original SFS database in any way. Therefore, it is safe to revert to the HPSS code, HPSS binaries and the SFS databases back to version 4.5 up until the HPSS 6.2 system is started.



- *Once the 6.2 HPSS servers are running, if operations such as files creations or deletions, migration, purge, repack, or reclaim are performed, reverting the system back to the 4.5 metadata will result in corruption of user data*
- *If shelf tape operation was performed on the 6.2 system, reverting back to 4.5 may result*

in tape mount errors for read operations

Once it is determined that it's safe to revert back to Release 6.2, perform the steps as follows. *Note that this procedure assumes that the upgrade stopped after the 6.2 servers are up but before changes were made that may affect the 4.5 user data or invalidate the 4.5 metadata.*

Perform the following steps for each root and non-root subsystem:

1. Shutdown all HPSS servers, if running.
2. Shutdown DB2. This step is optional since it does not affect the 4.5 system.
3. Undo DCE changes. This step is optional since it does not affect the 4.5 system.

```
% db_convert_dce_cds_undo -sub <subsys id> \  
-g /./:/encina/sfs/hpss/globalconfig
```
4. Restore the rc.hpss script in /etc. Perform this step on each HPSS machine (e.g. remote movers).

```
% cd /etc  
% mv rc.hpss_45 rc.hpss
```
5. Restore the FTP config, assuming that the PFTP Daemon is to run on the root subsystem.
Restore the /etc/inetd.conf file

```
% cd /etc  
% mv inetd.conf_45 inetd.conf
```


Refresh the inetd Daemon

```
% refresh -s inetd, or  
% kill -1 <inetd pid>
```
6. Restore the HPSS 4.5 /var/hpss directory and its contents

```
% cd /var  
% rm -Rf hpss  
% mv hpss_45 hpss
```
7. Restore the HPSS server keytab files

```
% cd /krb5  
% mv hpss.keytabs_45 hpss.keytabs
```
8. Restore the installed HPSS 4.5 code. Note that lpp version for the HPSS installation is 6.2 and not 4.5

```
% cd /opt  
% mv hpss hpss_62  
% mv hpss_45 hpss
```
9. Start HPSS 4.5 servers

6.3.24.2. Revert the HPSS 6.2 System to Version 5.1

Reverting to HPSS 5.1 from HPSS 6.2 is accomplished by renaming the tables in each DB2 database (e.g. CFG and SUBSYS1) that begin with "PRE62_" to their original name; the same name without

the "PRE62_". Next, beginning with step 4, follow the procedures in Section 6.3.24.1: *Revert the HPSS 6.2 System to Version 4.5* on page 225. The site will also have to revert the operating system software and any other upgraded software back to the HPSS 4.5 prerequisite versions so that the HPSS 4.5 code is not running on unsupported software.

6.4. Metadata Conversion Troubleshooting Procedures

This section provides troubleshooting procedures for common errors that may occur during a metadata conversion from HPSS 4.5 or HPSS 5.1 to HPSS 6.2. This file does not contain all error messages that a failed conversion might output but does discuss the more common errors.

If a DB2 or SQL error message is not found in this section, use DB2's CLI to obtain more information about the error. For example:

```
$ db2 ? SQL0100W
```

This will show more details about SQL warning message 100. Another reference that will help in determining actions upon receiving DB2 or SQL error messages, is the DB2 Message Reference. This guide has a list of every message DB2 can produce and recommended actions for fixing the problem.

In the following sections, the error text is displayed in bold type followed by a "=>" then an explanation of the error. The corresponding "Resolution:" paragraph(s) detail the recommended course of action to correct the problem and then continue with the conversion.

6.4.1. HPSS 4.5 to 6.2 Conversion Utility Errors and Warnings

This section provides troubleshooting procedures for errors that occur during a metadata conversion from 4.5 to 6.2. This file does not contain all error messages that a failed conversion might output but does provide some of the more common errors a user might see. It is helpful to keep in mind that the SFS (HPSS 4.5) metadata is only being read and then modified in memory and inserted into a DB2 table. So, even with the worst of error conditions, a user could empty all DB2 tables using the `db*_convert_empty` utilities provided, and start the conversion over from scratch with no harm to the metadata, DB2 or SFS.I

6.4.1.1. db_convert_collect_info Errors

Error: Could not open file /var/hpss/convert/6.2/Convert.DB.Names => The program was unable to open a text file for writing in /var/hpss/convert/6.2 directory.

Resolution: Check to make sure the directory exists and that write permissions are set properly.

Error: DB2 Database Descriptive Name must be no more than 32 characters long. => The descriptive name entered for the DB that is longer than the allowed 32 character string size.

Resolution: Rerun `db_convert_collect_info` and choose a name that is no more than 32 characters long.

Error: Must enter different DB Name for each subsystem. => The same database name for multiple subsystems was entered. Only one subsystem can share the same database as the global database, all other subsystems must have their own database.

Resolution: Create a different database, each with a unique database name, for each subsystem to be converted. Rerun the `db_convert_collect_info` utility and provide a unique database name for each subsystem.

6.4.1.2. db_config_convert, db_subsys_convert, and db_lr_convert Errors and Warnings

1SQL3304N The table does not exist. => The program is attempting to insert data into a table that has not been created under the database name and schema it is running under. Determine the database name entered for the subsystem during the db_convert_collect_info program (i.e. Suppose “dwcenrl” was entered for the database name for subsystem 1; however, the table was created under a different database name).

Resolution: DB2 associates your moniker with the schema name; ensure that tables are created under the schema name that is the same as the moniker you are running the program as (i.e. if you are 'hpss', make sure the tables are created under schema 'hpss' in db2). Also ensure that the correct database name is entered while running db_convert_collect_info and check that the table exists in that database.

awk: 0602-533 Cannot find or open file /var/hpss/convert/6.2/Convert.DB.Names. => The program cannot find the text files that resulted from running the collect_info utilities.

Resolution: Ensure that collect_info utilities is run prior to running any conversion programs. Also, look at db_config_convert, db_subsys_convert, and db_lr_convert to ensure the path to those text files has not changed (default is /var/hpss/convert/6.2).

sfs_OpenFile: ENC-sfs-0034: Insufficient privilege for this operation. => You do not have the proper DCE credentials.

Resolution: Log into DCE with the proper credentials to allow an SFS read and rerun the conversion program.

Error: sfs_OpenFile: ENC-sfs-0051: Unknown file system name. => The program cannot find your DCE credentials or the credentials that you have do not allow read access to SFS files.

Resolution: Log into DCE with the proper credentials to allow an SFS read and rerun the conversion program.

1SQL3805N The state of the application or of one or more table spaces for the table specified prohibits the loadapi action or quiescemode "2". Reason code = "1". => The DB2 database is probably stuck in “load pending” mode from a previous run of a long running utility that exited with an error. Or the program is trying to insert data into multiple DB2 tables (i.e. db_convert_nsubject) that share the same tablespace (i.e. NSACL, and NSOBJECT).

Resolution: Either use the restart procedues (see Section 6.3.13.5: *Restarting the Long Running Utilities* on page 207) or drop the table from DB2 and then recreate the table and run the utility again. Or if the db_convert_nsubject utility is running, ensure that both the NSOBJECT and NSACL tables are in different tablespaces because DB2 places an exclusive lock on each tablespace it is loading into.

Error: hpss.sspvtape table could not be emptied properly. => The table listed where 'hpss' is the schema name and 'sspvtape' is the name of the table, could not be emptied.

Resolution: Check that the table exists in the schema and that the `db_convert_collect_info` utility has been run recently. The empty table scripts pull their information about which schema to use from the text file `/var/hpss/convert/6.2/Convert.DB.Names`, created by running `db_convert_collect_info`. Enter the appropriate schema name for the global database (which will be used for all databases) when rerunning `db_convert_collect_info`. This could also mean that the tablespace that the unemptiable tables exist under has quiesced to EXCLUSIVE mode (i.e. there is a lock on the tablespace). Look at the tablespaces using the DB2 control center utility or issue “list tablespaces” on the DB2 CLI. If the tablespace mode is EXCLUSIVE, use the CLI to issue the “quiesce tablespaces for table **** reset” for each table that exists in the tablespace to free the lock and rerun the conversion program.

An error occurred, not all subsystem tables emptied in global, under schema hpss. => This error is explaining that an error has occurred on one or more 'delete from ????' operations, where '????' is a schema and table name.

Resolution: Verify that the table exists in the schema and that the `db_convert_collect_info` utility has been run recently. The empty table scripts pull their information about which schema to use from the text file `/var/hpss/convert/6.2/Convert.DB.Names`, created by running `db_convert_collect_info`. Enter the appropriate schema name for the global database (which will be used for all databases) when rerunning `db_convert_collect_info`. This could also mean that the tablespace that the unemptiable tables exist under has quiesced to EXCLUSIVE mode (i.e. there is a lock on the tablespace). Look at the tablespaces using db2's control center utility or issue “list tablespaces” on the DB2 CLI. If the tablespace mode is EXCLUSIVE, use the CLI to issue the “quiesce tablespaces for table **** reset” for each table that exists in the tablespace to free the lock and rerun the conversion program.

Error: Cannot open file '/var/hpss/convert/6.2/sample' => The conversion program cannot access the text file that maps DB Names to SubsysIds, or SSIIds to CoreIds, or NSIds to CoreIds.

Resolution: Verify that the pathname of the text file that it is trying to access hasn't been modified in the running script. If not, rerun the `db_convert_collect_info` utility because the text file was never created or has been modified.

Error: Key '8cf7-754d66-7652cb3-21' not found in /var/hpss/convert/6.2/Convert.SS.Server.Ids => In this case, the conversion program is attempting to map an HPSS 4.5 Storage Server id to a new HPSS 6.2 Core Server id, and it cannot find an entry in the text file, `Convert.SS.Server.Ids`, under `/var/hpss/convert/6.2`. This text file is created after running `db_convert_collect_info` and results from reading the general and specific config entries for all storage servers in the HPSS 4.5 system. Therefore, if both a general and specific configuration record for this storage server does not exist, either create the storage server this conversion program is trying to find, or delete the offending metadata (i.e. orphaned storage segments). The conversion programs have a force option (`-force`) that allows the conversion programs to output an error and continue the conversion process after finding orphaned metadata entries; however, we strongly recommend that you contact your HPSS customer support representative before using the force option, as converting essential metadata could be skipped.

Resolution: Fix the HPSS configuration for the incomplete or missing 4.5 server or contact your HPSS customer support representative and run the conversion with the force option to skip possible orphaned metadata entries.

Error: /.:/encina/sfs/hpss/nsubjects is not in subsystem 1 => The program is attempting to convert the "nsubjects" SFS file and discovered that the filename differs from your HPSS systems specific configuration file `NSObjFileName` for subsystem 1.

Resolution: Either the wrong filename or the wrong subsystem Id for that filename was entered. Rerun the conversion program with the correct subsystem Id and filename.

SQL0968C The file system is full. SQLSTATE=57011 => This error message usually follows 10 or so other SQL messages, and means that the load operation for the long running conversion program failed because, most likely, the system temporary tablespace (which defaults to TEMPSPACE1) has run out of room. Or it could mean that the tablespace that was being written to is full (an SMS or system managed tablespace).

Resolution: Either add a container to TEMPSPACE1, or define a new tablespace of type system temporary with a new container (new device) to expand the available system temporary space for the database. Recycle the database to enable use of the newly defined system temporary tablespace and rerun the conversion program. Or add a container to the tablespace where data was being written and rerun the conversion program following restart procedures in Section 6.3.13.5: *Restarting the Long Running Utilities* on page 207.

SQL0289N Unable to allocate new pages in table space "BITFILESMALLSPACE". SQLSTATE=57011 => The tablespace "BITFILESMALLSPACE" is full. It is slightly different than the error above because BITFILESMALLSPACE is a DMS (database managed) container or raw device.

Resolution: Add a container to the tablespace or extend the size of the existing container and rerun the conversion program using restart procedures in Section 6.3.13.5: *Restarting the Long Running Utilities* on page 207.

Error: mm_InsertRecord failed, mm_status = -2001, for entry -5590 => This error translates to an mmlib HPSS_E_MM_DB_ERROR, which could mean several things. Most likely the program was trying to insert data into a table that does not exist. But the table could also exist in a locked tablespace (i.e., in a load pending state).

Resolution: Ensure the table exists for the running conversion program. Also check the tablespaces to ensure the state is Normal, and not Quiesced, or Load Pending. If the tablespace is not in a Normal state, then ensure that no other conversion programs are running. If the tablespace should not be in a Load Pending or Quiesced state, then recycle the database. Finally, run the conversion program again.

Error: mm_InsertRecord failed, mm_status = -2014, for entry 0 => This error translates to an mmlib HPSS_E_MM_DUPLICATE error which means that the table, where the data insert was attempted, already has data. If this error appears while running db_config_convert or db_subsys_convert, it is likely that the db_config_convert_empty or db_subsys_convert_empty script failed (the tables are emptied before running the db_config_convert and db_subsys_convert programs).

Resolution: Use the db2 CLI (type 'db2') and connect to the database where the records are to be inserted, then do a 'select * from ????' where '????' is the table where the records are to be inserted. If records exist, then it's likely that the transaction log is full (i.e., an attempt was made to delete lots of records but the transaction log isn't big enough). If the transaction log is full, then increase the size of LOGFILSIZ (database configuration parameter). If this is already set at the maximum value, then try increasing the number of primary log files (database configuration parameter). Ensure sufficient log file or disk space (that these database configuration parameters are not over allocated when there isn't enough space). Recycle the database after adjusting the database configuration parameters mentioned above and rerun the conversion program.

Error: write_bitfile_to_ASC failed to write to pipe, fd is -1, errno = 9 => This error results when running a long running conversion program that cannot write to /var/hpss/convert/6.2 and create a file called asc_fifo_bitfile (the named pipe for the C program to communicate with the DB2 load).

Resolution: Ensure that the user running the conversion has write permissions on /var/hpss/convert/6.2. Rerun the conversion program.

Could not find database 'subsys1' using schema 'hpss'. => This error is usually followed by **Error: mm_InsertRecord failed, mm_status = -2001, for entry 0** and indicates that the program could not connect to the database.

Resolution: Make sure the database and schema exist (and that the schema is the same as the user under which the conversion is being run). For example, when trying to insert data into a table under schema 'hpss', run the conversion as user 'hpss'. Rerun the conversion program. Sometimes alterations to the database, or starting and stopping DB2, can result in a first connection failing; normally the second connection attempt succeeds.

Error: MigPolID 1 is in use by both disk and tape storage classes, exiting! => The db_convert_migpol, or migration policy conversion, program produces this error upon trying to convert a migration policy (in this case, migration policy 1) that in 4.5 was used by both disk and tape storage classes. This can no longer be the case as of HPSS version 5.1.

Resolution: Create a new identical migration policy and set the disk storage class to use one policy and the tape storage class to use the other. Rerun the conversion program.

Error: convert_ReadServer failed, mm_status = -32016 => This error results from a partially configured 4.5 system. Specifically, the conversion program found a general configuration record for a Storage Server or Name Server and could not find the corresponding specific configuration record.

Resolution: Either remove the general configuration record for the partially configured server or contact your HPSS customer support representative and use the force option to skip converting metadata related to the partially configured server.

6.4.2. HPSS 5.1 to 6.2 Conversion Utility Errors

This section provides troubleshooting procedures for errors that occur during the metadata conversion from HPSS 5.1 to 6.2.

6.4.2.1. hpss_md_convert_51 Errors

Description of Error: When running option 5 (Convert Server Generic config table) Error -2 is HPSS_ENOENT on call to mm_GetDefaultServerConfig(), <no error> is encountered.

Resolution: The uuid_create call can fail if /var/hpss/etc/ieee_802_addr file doesn't exist. Run mkhpss and select the option to create the ieee_802_addr file. Next, rerun the hpss_md_convert_51 program and select option 5 again.

Description of Error: Any error with SQL0803N error in its description.

Resolution: This error occurs when there is already metadata in the table where the conversion is

trying to insert the converted metadata. If this resulted from previously running the metadata conversion, and the metadata should be re-converted, then identify the table name (e.g. “HPSS.SERVERINTERFACES”) and empty the table by executing a

```
% db2 delete from <table name>
```

statement where <table name> is the exact table name in the statement (e.g. HPSS.SERVER). If the conversion for this table has not been previously, then contact your HPSS customer support representative before deleting the metadata as the metadata may be necessary for HPSS operation.

6.4.2.2. hpss_init_server_acls Errors

Description of Error: The program provides the following error output:

```
% hpss_init_server_acls
Error in stat of Keytab File, /var/hpss/etc/mm.keytab, 2
connect: A remote host refused an attempted connect operation.
Failed to get default ACL:
<no error>
```

Resolution: This error implies that the program cannot contact the site’s configured authorization mechanism as specified in /var/hpss/etc/site.conf. Check the entry for this hostname in the site.conf file and ensure that the mechanism is functioning correctly. In tests, this usually indicated that LDAP was not running, and the problem was resolved by simply starting LDAP on the test system.

6.5. HPSS 4.5 Conversion Utilities Output

6.5.1. Interpreting Output from the 4.5 Conversion Utility

Generally, the following output is displayed when running a configuration or subsystem conversion program from the command line:

```
Running lr_db_convert_storagesegdisk utility...
  Converting SFS file /./encina/sfs/hpss/storagesegdisk.1
```

The output indicates the name of the utility that is running and the name of the SFS file that it is currently converting. If the program exits with an error, take action to correct the error and rerun the conversion program again. Note that the conversion programs for configuration and subsystem metadata, `db_config_convert` and `db_subsys_convert`, will automatically empty all tables in the database; however, you can run `db_config_convert_empty` or `db_subsys_convert_empty` which will also empty the tables.

For the long running programs, you will see:

```
Running lr_db_convert_nsobject utility...
  Converting SFS file /./encina/sfs/hpss/nsobjects.
```

The output indicates the name of the utility that is running and the name of the SFS file that it is currently converting. The program will pause after displaying this output so that it can perform the conversion. It will output a “.” for every 10,000 records it has converted and a number enclosed in brackets for every 500,000 records it has converted. Note, the “.” does not represent the number of records processed or the number of records in the SFS file, it only represents the number of records actually converted.

If the program appears to hang for longer than two minutes and it is killed (by ^C), there is a chance that DB2 will be placed in an inconsistent state. Evidence of a DB2 inconsistency will be seen either

through several syslogd error messages output to the monitor or a failed attempt at restarting the failed conversion program. Should DB2 be placed in an inconsistent state, the tablespace will likely have to be dropped and recreated along with the table being converted. Then the database will need to be recycled (using db2stop and db2start).

Note that the long running conversion programs are started from a script that will call a C program to read metadata from SFS, convert the data, and write the data to a named pipe. The script then issues the DB2 load command which reads from the named pipe. To ensure correctness of the metadata converted, the number of records submitted to the load from the C program should equal the number of committed rows by the load utility. Output for a successful conversion will look like the following:

```
Running lr_db_convert_nsubject utility...
  Converting SFS file
././encina/sfs/hpss/nsubjects.1.....
  Converted 103749 records successfully from
././encina/sfs/hpss/nsubjects.1
lr_db_convert_nsubject complete, submitted 0 records to DB2 nsacl
load for subsystem 1
lr_db_convert_nsubject complete, submitted 103749 records to DB2
nsubject load for subsystem 1, 1870 operations per sec, 55.455049
total time
```

```
Number of rows read          = 103749
Number of rows skipped       = 0
Number of rows loaded        = 103749
Number of rows rejected      = 0
Number of rows deleted       = 0
Number of rows committed    = 103749
```

Notice that the number of rows committed by the load (103749) is equal to the C program's number of rows submitted to the load (103749). If the numbers do not agree or any deleted, skipped or rejected rows are encountered, the conversion program has not converted the metadata correctly. In this case, contact your HPSS customer support representative.

In the case that you have already run a long running conversion and it failed, follow restart procedures in Section 6.3.13.5: *Restarting the Long Running Utilities* on page 207. The output upon restart should look like the following:

```
Re-running lr_db_convert_nsubject utility...
  Converting SFS file ././encina/sfs/hpss/nsubjects.1.....
  Converted 103749 records successfully from
././encina/sfs/hpss/nsubjects.1
lr_db_convert_nsubject complete, submitted 0 records to DB2 nsacl
load for subsystem 1
lr_db_convert_nsubject complete, submitted 103749 records to DB2
nsubject load for subsystem 1, 1870 operations per sec, 55.455049
total time
```

```
Number of rows read          = 103749
Number of rows skipped       = 0
Number of rows loaded        = 103749
Number of rows rejected      = 0
Number of rows deleted       = 0
Number of rows committed    = 103749
```

The output displays “Re-running” which indicates that the utility is performing a restart. When performing a restart and there are already rows in the DB2 table, rows committed should still equal rows submitted to the load. The conversion program will determine the number of records already loaded into the table, find the next consecutive SFS record to load, and continue counting until all SFS records are converted properly and inserted into the DB2 table(s).

In order to accomplish a successful conversion, it is recommended that you do not run any programs individually from the command line with the -f option. The -f option will only convert the specified file. Using the -f option is undesirable when converting all cartridges in a system since using this option only converts the specific cartridge file specified on the command line. The -f option is only intended for debugging purposes and should only be used after consulting with your HPSS customer support representative.

6.5.2. Examples of HPSS 4.5 Conversion Utility Output

6.5.2.1. db_convert_collect_info Output

Below is the expected output on a successful run of db_convert_collect_info, the collect information program. The system being converted has two subsystems and all the tables needed in DB2 have already created in database “global”, “subsys1”, and “subsys2” under schema “hpss”.

```
/opt/hpss/bin> su - hpss
$ db_convert_collect_info /./encina/sfs/hpss/globalconfig
Logged into DCE as hpss_ssm.
Running db_convert_prompt_db_names utility...
  For Global DB, enter the DB Name: global
  For Global DB, enter the Schema Name: hpss
  For Global DB, enter the Descriptive Name: test db
  For subsystem 1, enter the DB Name: subsys1
  For subsystem 2, enter the DB Name: subsys2
  Created file /var/hpss/convert/6.2/Convert.DB.Names
db_convert_prompt_db_names complete, DBName text file created.

Running db_convert_get_ns_server_ids utility...
  Created file /var/hpss/convert/6.2/Convert.NS.Server.Ids
db_convert_get_ns_server_ids complete, NSId text file created.

Running db_convert_get_ss_server_ids utility...
  Created file /var/hpss/convert/6.2/Convert.SS.Server.Ids
db_convert_get_ss_server_ids complete, SSId text file created.

Acquired DCE credentials have been destroyed.

db_convert_collect_info complete.
$
```

6.5.2.2. db_config_convert Output

Below is the expected output on a successful run of db_config_convert, the configuration metadata conversion utility.

```

$ db_config_convert ../encina/sfs/hpss/globalconfig

Logged into DCE as hpss_ssm.

Emptying all configuration tables in global database...

All config tables emptied in global database, under schema hpss.

Running db_convert_global utility...
  Converting SFS file ../encina/sfs/hpss/globalconfig
  Converted 1 record successfully from
../encina/sfs/hpss/globalconfig
db_convert_global complete, inserted 1 record into DB2 global table

Running db_convert_storsubsys utility...
  Converting SFS file ../encina/sfs/hpss/storsubsysconfig
  Converted 2 records successfully from
../encina/sfs/hpss/storsubsysconfig
db_convert_storsubsys complete, inserted 2 records into DB2
storsubsys table
db_convert_storsubsys complete, inserted 13 records into DB2
subsyscos table

Running db_convert_server utility...
  Converting SFS file ../encina/sfs/hpss/serverconfig
  Converted 18 records successfully from
../encina/sfs/hpss/serverconfig
db_convert_server complete, inserted 18 records into DB2 server
table

Running db_convert_core utility...
  Making Core Server for Subsystem 1
  Converted BFS, NS, and SS into Core Server for subsystem 1
  Making Core Server for Subsystem 2
  Converted BFS, NS, and SS into Core Server for subsystem 2
db_convert_core complete, inserted 2 records into DB2 core table

Running db_convert_dmg utility...
  The local SFS system has no DMG servers to convert.
db_convert_dmg complete, inserted 0 records into DB2 dmg table

Running db_convert_acctcfg utility...
  Converting SFS file ../encina/sfs/hpss/accounting
  Converted 1 records successfully from
../encina/sfs/hpss/accounting
db_convert_acctcfg complete, inserted 1 records into DB2 acctcfg
table

Running db_convert_acctval utility...
  Converting SFS file ../encina/sfs/hpss/acctvalidate
  Converted 18 records successfully from
../encina/sfs/hpss/acctvalidate
db_convert_acctval complete, inserted 18 records into DB2 acctval
table

```

```
Running db_convert_gatekeeper utility...
    Converting SFS file ../encina/sfs/hpss/gkconfig
    Converted 1 records successfully from
../encina/sfs/hpss/gkconfig
db_convert_gatekeeper complete, inserted 1 records into DB2
gatekeeper table

Running db_convert_logclient utility...
    Converting SFS file ../encina/sfs/hpss/logclient
    Converted 1 records successfully from
../encina/sfs/hpss/logclient
db_convert_logclient complete, inserted 1 records into DB2 logclient
table

Running db_convert_logdaemon utility...
    Converting SFS file ../encina/sfs/hpss/logdaemon
    Converted 1 records successfully from
../encina/sfs/hpss/logdaemon
db_convert_logdaemon complete, inserted 1 records into DB2 logdaemon
table

Running db_convert_lspolicy utility...
    Converting SFS file ../encina/sfs/hpss/lspolicy
    Converted 1 records successfully from
../encina/sfs/hpss/lspolicy
db_convert_lspolicy complete, inserted 1 records into DB2 lspolicy
table

Running db_convert_mps utility...
    Converting SFS file ../encina/sfs/hpss/mps
    Converted 1 records successfully from ../encina/sfs/hpss/mps
db_convert_mps complete, inserted 1 records into DB2 mps table

Running db_convert_ndcg utility...
    The local SFS system has no NDCG servers to convert.
db_convert_ndcg complete, inserted 0 records into DB2 ndcg table

Running db_convert_nfs utility...
    Converting SFS file ../encina/sfs/hpss/nfs
    Converted 1 records successfully from ../encina/sfs/hpss/nfs
db_convert_nfs complete, inserted 1 records into DB2 nfs table

Running db_convert_pvl utility...
    Converting SFS file ../encina/sfs/hpss/pvl
    Converted 1 records successfully from ../encina/sfs/hpss/pvl
db_convert_pvl complete, inserted 1 records into DB2 pvl table

Running db_convert_pvr utility...
    Converting SFS file ../encina/sfs/hpss/pvr
    Converted 2 records successfully from ../encina/sfs/hpss/pvr
db_convert_pvr complete, inserted 2 records into DB2 pvr table

Running db_convert_cartridge utility...
    Converting SFS file ../encina/sfs/hpss/cartridge_stk
    Converted 28 records successfully from
```

```

./../encina/sfs/hpss/cartridge_stk
    Converting SFS file ./../encina/sfs/hpss/cartridge_stk_rait
    Converted 4 records successfully from
./../encina/sfs/hpss/cartridge_stk_rait
db_convert_cartridge complete, inserted 32 records into DB2
cartridge table

Running db_convert_cos utility...
    Converting SFS file ./../encina/sfs/hpss/cos
    Converted 21 records successfully from ./../encina/sfs/hpss/cos
db_convert_cos complete, inserted 21 records into DB2 cos table

Running db_convert_dmgfileset utility...
    The local SFS system has no DMG Filesets to convert.
db_convert_dmgfileset complete, inserted 0 records into DB2
dmgfileset table

Running db_convert_filefamily utility...
    Converting SFS file ./../encina/sfs/hpss/filefamily
    Converted 0 records successfully from
./../encina/sfs/hpss/filefamily
db_convert_filefamily complete, inserted 0 records into DB2
filefamily table

Running db_convert_hier utility...
    Converting SFS file ./../encina/sfs/hpss/hierarchy
    Converted 21 records successfully from
./../encina/sfs/hpss/hierarchy
db_convert_hier complete, inserted 21 records into DB2 hier table

Running db_convert_migpol utility...
    Converting SFS file ./../encina/sfs/hpss/migpolicy
    Converted 4 records successfully from
./../encina/sfs/hpss/migpolicy
db_convert_migpol complete, inserted 4 records into DB2 migpol table

Running db_convert_mover utility...
    Converting SFS file ./../encina/sfs/hpss/mover
    Converted 3 records successfully from ./../encina/sfs/hpss/mover
db_convert_mover complete, inserted 3 records into DB2 mover table

Running db_convert_moverdevice utility...
    Converting SFS file ./../encina/sfs/hpss/moverdevice
    Converted 17 records successfully from
./../encina/sfs/hpss/moverdevice
db_convert_moverdevice complete, inserted 17 records into DB2
moverdevice table

Running db_convert_nsglobalfileset utility...
    Converting SFS file ./../encina/sfs/hpss/nsglobalfilesets
    Converted 2 records successfully from
./../encina/sfs/hpss/nsglobalfilesets
db_convert_nsglobalfileset complete, inserted 2 records into DB2
nsglobalfileset table

```

```

Running db_convert_purgepol utility...
    Converting SFS file /./encina/sfs/hpss/purgepolicy
    Converted 1 records successfully from
././encina/sfs/hpss/purgepolicy
db_convert_purgepol complete, inserted 1 records into DB2 purgepol
table

Running db_convert_pvldrive utility...
    Converting SFS file /./encina/sfs/hpss/pvldrive
    Converted 17 records successfully from
././encina/sfs/hpss/pvldrive
db_convert_pvldrive complete, inserted 17 records into DB2 pvldrive
table

Running db_convert_pvlpv utility...
    Converting SFS file /./encina/sfs/hpss/pvlpv
    Converted 36 records successfully from /./encina/sfs/hpss/pvlpv
db_convert_pvlpv complete, inserted 36 records into DB2 pvlpv table

Running db_convert_sclassthreshold utility...
    Converting SFS file /./encina/sfs/hpss/sclassthreshold
    Converted 1 records successfully from
././encina/sfs/hpss/sclassthreshold
db_convert_sclassthreshold complete, inserted 1 records into DB2
sclassthreshold table

Running db_convert_logpolicy utility...
    Converting SFS file /./encina/sfs/hpss/logpolicy
    Converted 12 records successfully from
././encina/sfs/hpss/logpolicy
db_convert_logpolicy complete, inserted 12 records into DB2
logpolicy table

Running db_convert_site utility...
    Converting SFS file /./encina/sfs/hpss/site
    Converted 0 records successfully from /./encina/sfs/hpss/site
db_convert_site complete, inserted 0 records into DB2 site table

Running db_convert_sclass utility...
    Converting SFS file /./encina/sfs/hpss/storageclass
    Converted 12 records successfully from
././encina/sfs/hpss/storageclass
db_convert_sclass complete, inserted 12 records into DB2 sclass
table

Acquired DCE credentials have been destroyed.

db_config_convert complete.
$

```

6.5.2.3. db_subsys_convert Output

Below is the expected output on a successful run of db_subsys_convert, the subsystem metadata conversion utility.

```
$ db_subsys_convert /./encina/sfs/hpss/globalconfig 1

Logged into DCE as hpss_ssm.

Emptying all subsystem tables in subsys1 database...

All subsystem tables emptied in subsys1 database, under schema hpss.

Running db_convert_acctlog utility...
  Converting SFS file /./encina/sfs/hpss/acctlog.1
  Converted 0 records successfully from
/./encina/sfs/hpss/acctlog.1
db_convert_acctlog complete, inserted 0 records into DB2 acctlog
table for subsystem 1

Running db_convert_acctsnap utility...
  Converting SFS file /./encina/sfs/hpss/acctsnap
  Converted 0 records successfully from
/./encina/sfs/hpss/acctsnap
  Converting SFS file /./encina/sfs/hpss/acctsnap
  Converted 0 records successfully from
/./encina/sfs/hpss/acctsnap
db_convert_acctsnap complete, inserted 0 records into DB2 acctsnap
table for subsystem 1

Running db_convert_acctsum utility...
  Converting SFS file /./encina/sfs/hpss/acctsum
  Converted 242 records successfully from
/./encina/sfs/hpss/acctsum
db_convert_acctsum complete, inserted 242 records into DB2 acctsum
table for subsystem 1

Running db_convert_bfcoschange utility...
  Converting SFS file /./encina/sfs/hpss/bfcoschange.1
  Converted 0 records successfully from
/./encina/sfs/hpss/bfcoschange.1
db_convert_bfcoschange complete, inserted 0 records into DB2
bfcoschange table for subsystem 1

Running db_convert_bfdiskallocrec utility...
  Converting SFS file /./encina/sfs/hpss/bfdiskallocrec.1
  Converted 1417 records successfully from
/./encina/sfs/hpss/bfdiskallocrec.1
db_convert_bfdiskallocrec complete, inserted 1417 records into DB2
bfdiskallocrec table for subsystem 1

Running db_convert_bfdiskseg utility...
  Converting SFS file /./encina/sfs/hpss/bfdisksegment.1
  Converted 1352 records successfully from
/./encina/sfs/hpss/bfdisksegment.1
db_convert_bfdiskseg complete, inserted 1352 records into DB2
bfdiskseg table for subsystem 1

Running db_convert_bfmigrec utility...
  Converting SFS file /./encina/sfs/hpss/bfmigrrec.1
```

```
Converted 600 records successfully from
././encina/sfs/hpss/bfmigrrec.1
db_convert_bfmigrrec complete, inserted 600 records into DB2 bfmigrrec
table for subsystem 1
```

```
Running db_convert_bfpurgerec utility...
  Converting SFS file ././encina/sfs/hpss/bfpurgerec.1
  Converted 0 records successfully from
././encina/sfs/hpss/bfpurgerec.1
db_convert_bfpurgerec complete, inserted 0 records into DB2
bfpurgerec table for subsystem 1
```

```
Running db_convert_bfssegchkpt utility...
  Converting SFS file ././encina/sfs/hpss/bfssegchkpt.1
  Converted 0 records successfully from
././encina/sfs/hpss/bfssegchkpt.1
db_convert_bfssegchkpt complete, inserted 0 records into DB2
bfssegchkpt table for subsystem 1
```

```
Running db_convert_bfssunlink utility...
  Converting SFS file ././encina/sfs/hpss/bfssunlink.1
  Converted 0 records successfully from
././encina/sfs/hpss/bfssunlink.1
db_convert_bfssunlink complete, inserted 0 records into DB2
bfssunlink table for subsystem 1
```

```
Running db_convert_nsacl utility...
  Converting SFS file ././encina/sfs/hpss/nsacls.1
  Converted 0 records successfully from
././encina/sfs/hpss/nsacls.1
db_convert_nsacl complete, inserted 0 records into DB2 nsacl table
for subsystem 1
```

```
Running db_convert_nsfilesetattr utility...
  Converting SFS file ././encina/sfs/hpss/nsfilesetattrs.1
  Converted 1 records successfully from
././encina/sfs/hpss/nsfilesetattrs.1
db_convert_nsfilesetattr complete, inserted 1 records into DB2
nsfilesetattr table for subsystem 1
```

```
Running db_convert_sspvdisk utility...
  Converting SFS file ././encina/sfs/hpss/sspvdisk.1
  Converted 3 records successfully from
././encina/sfs/hpss/sspvdisk.1
db_convert_sspvdisk complete, inserted 3 records into DB2 sspvdisk
table for subsystem 1
```

```
Running db_convert_sspvtape utility...
  Converting SFS file ././encina/sfs/hpss/sspvtape.1
  Converted 31 records successfully from
././encina/sfs/hpss/sspvtape.1
  Converting SFS file ././encina/sfs/hpss/sspvtape.1.2
  Converted 0 records successfully from
././encina/sfs/hpss/sspvtape.1.2
db_convert_sspvtape complete, inserted 31 records into DB2 sspvtape
```


table for subsystem 1

Running db_convert_storagemapdisk utility...

```
    Converting SFS file /./encina/sfs/hpss/storagemapdisk.1
    Converted 3 records successfully from
././encina/sfs/hpss/storagemapdisk.1
db_convert_storagemapdisk complete, inserted 3 records into DB2
storagemapdisk table for subsystem 1
db_convert_storagemapdisk complete, inserted 1 records into DB2
diskspace table for subsystem 1
```

Running db_convert_storagemaptape utility...

```
    Converting SFS file /./encina/sfs/hpss/storagemaptape.1
    Converted 30 records successfully from
././encina/sfs/hpss/storagemaptape.1
    Converting SFS file /./encina/sfs/hpss/storagemaptape.1.2
    Converted 0 records successfully from
././encina/sfs/hpss/storagemaptape.1.2
db_convert_storagemaptape complete, inserted 30 records into DB2
storagemaptape table
for subsystem 1
```

Running db_convert_vvdisk utility...

```
    Converting SFS file /./encina/sfs/hpss/vvdisk.1
    Converted 3 records successfully from
././encina/sfs/hpss/vvdisk.1
db_convert_vvdisk complete, inserted 3 records into DB2 vvdisk table
for subsystem 1
```

Running db_convert_vvtape utility...

```
    Converting SFS file /./encina/sfs/hpss/vvtape.1
    Converted 22 records successfully from
././encina/sfs/hpss/vvtape.1
    Converting SFS file /./encina/sfs/hpss/vvtape.1.2
    Converted 0 records successfully from
././encina/sfs/hpss/vvtape.1.2
db_convert_vvtape complete, inserted 22 records into DB2 vvtape
table for subsystem 1
```

Acquired DCE credentials have been destroyed.

db_subsys_convert complete.

\$

6.5.2.4. Long Running Conversion Utilities Output

Output for a successful long running conversion is shown below. Note that it is possible for the output from the long running conversion program to be out of order. If a conversion program finishes and DB2 is busy building indexes for a table, the conversion script will move to the next conversion program and start output for the new program before the DB2 output is displayed. This is normal and cannot be controlled.

Running lr_db_convert_bftapeseg utility...

Converting SFS file /./encina/sfs/hpss/bftapesegment.1
Converted 1159 records successfully from
/./encina/sfs/hpss/bftapesegment.1
lr_db_convert_bftapeseg complete, submitted 1159 records to DB2
bftapeseg load for subsystem 1, 1596 operations per sec, 0.725914
total time

Number of rows read = 1159
Number of rows skipped = 0
Number of rows loaded = 1159
Number of rows rejected = 0
Number of rows deleted = 0
Number of rows committed = 1159

Running lr_db_convert_bitfile utility...

Converting SFS file /./encina/sfs/hpss/bitfile.1..
Converted 28063 records successfully from
/./encina/sfs/hpss/bitfile.1
lr_db_convert_bitfile complete, submitted 28063 records to DB2
bitfile load for subsystem 1, 743 operations per sec, 37.721626
total time

Number of rows read = 28063
Number of rows skipped = 0
Number of rows loaded = 28063
Number of rows rejected = 0
Number of rows deleted = 0
Number of rows committed = 28063

Running lr_db_convert_nsubject utility...

Converting SFS file
/./encina/sfs/hpss/nsubjects.....
Converted 28819 records successfully from
/./encina/sfs/hpss/nsubjects.1
lr_db_convert_nsubject complete, submitted 0 records to DB2 nsacl
load for subsystem 1lr_db_convert_nsubject complete, submitted 28819
records to DB2 nsubject load for subsystem 1, 727 operations per
sec, 39.591100 total time

Number of rows read = 0
Number of rows skipped = 0
Number of rows loaded = 0
Number of rows rejected = 0
Number of rows deleted = 0
Number of rows committed = 0

Number of rows read = 28819
Number of rows skipped = 0
Number of rows loaded = 28819
Number of rows rejected = 0
Number of rows deleted = 0
Number of rows committed = 28819

Running lr_db_convert_nstext utility...

```
    Converting SFS file /./encina/sfs/hpss/nstext.1
    Converted 408 records successfully from
/./encina/sfs/hpss/nstext.1
lr_db_convert_nstext complete, submitted 408 records to DB2 nstext
load for subsystem 1, 1233 operations per sec, 1.001456 total time
```

```
Number of rows read           = 408
Number of rows skipped        = 0
Number of rows loaded         = 408
Number of rows rejected       = 0
Number of rows deleted        = 0
Number of rows committed     = 408
```

```
Running lr_db_convert_storagesegdisk utility...
```

```
    Converting SFS file /./encina/sfs/hpss/storagesegdisk.1
    Converted 1974 records successfully from
/./encina/sfs/hpss/storagesegdisk.1
lr_db_convert_storagesegdisk complete, submitted 1974 records to DB2
storagesegdisk load for subsystem 1, 986 operations per sec,
2.001883 total time
```

```
Number of rows read           = 1974
Number of rows skipped        = 0
Number of rows loaded         = 1974
Number of rows rejected       = 0
Number of rows deleted        = 0
Number of rows committed     = 1974
```

```
Running lr_db_convert_storagesegtape utility...
```

```
    Converting SFS file /./encina/sfs/hpss/storagesegtape.1
    Converted 1159 records successfully from
/./encina/sfs/hpss/storagesegtape.1
    Converting SFS file /./encina/sfs/hpss/storagesegtape.1.2
    Converted 0 records successfully from
/./encina/sfs/hpss/storagesegtape.1.2
lr_db_convert_storagesegtape complete, submitted 1159 records to DB2
storagesegtape load for subsystem 1, 53245 operations per sec,
0.021767 total time
```

```
Number of rows read           = 1159
Number of rows skipped        = 0
Number of rows loaded         = 1159
Number of rows rejected       = 0
Number of rows deleted        = 0
Number of rows committed     = 1159
```


Appendix A. Glossary of Terms and Acronyms

ACI	Automatic Media Library Client Interface
ACL	Access Control List
ACSL	Automated Cartridge System Library Software (Science Technology Corporation)
ADIC	Advanced Digital Information Corporation
accounting	The process of tracking system usage per user, possibly for the purposes of charging for that usage. Also, a log record message type used to log information to be used by the HPSS Accounting process. This message type is not currently used.
AIX	Advanced Interactive Executive. An operating system provided on many IBM machines.
alarm	A log record message type used to log high-level error conditions.
AML	Automated Media Library. A tape robot.
AMS	Archive Management Unit
ANSI	American National Standards Institute
API	Application Program Interface
archive	One or more interconnected storage systems of the same architecture.
attribute	When referring to a managed object, an attribute is one discrete piece of information, or set of related information, within that object.
attribute change	When referring to a managed object, an attribute change is the modification of an object attribute. This event may result in a notification being sent to SSM, if SSM is currently registered for that attribute.
audit (security)	An operation that produces lists of HPSS log messages whose record type is SECURITY. A security audit is used to provide a trail of security-relevant activity in HPSS.
bar code	An array of rectangular bars and spaces in a predetermined pattern which represent alphanumeric information in a machine readable format (e.g., UPC symbol)
BFS	HPSS Bitfile Service.
bitfile	A file stored in HPSS, represented as a logical string of bits unrestricted in size or internal structure. HPSS imposes a size limitation in 8-bit bytes based upon the maximum size in bytes that can be represented by a 64-bit unsigned integer.
bitfile segment	An internal metadata structure, not normally visible, used by the Core Server to map contiguous pieces of a bitfile to underlying storage.

Bitfile Service	Portion of the HPSS Core Server that provides a logical abstraction of bitfiles to its clients.
BMUX	Block Multiplexer Channel
bytes between tape marks	The number of data bytes that are written to a tape virtual volume before a tape mark is required on the physical media.
CAP	Cartridge Access Port
cartridge	A physical media container, such as a tape reel or cassette, capable of being mounted on and dismounted from a drive. A fixed disk is technically considered to be a cartridge because it meets this definition and can be logically mounted and dismounted.
central log	The main repository of logged messages from all HPSS servers enabled to send messages to the Log Daemon.
class	A type definition in Java. It defines a template on which objects with similar characteristics can be built, and includes variables and methods specific to the class.
Class of Service	A set of storage system characteristics used to group bitfiles with similar logical characteristics and performance requirements together. A Class of Service is supported by an underlying hierarchy of storage classes.
cluster	The unit of storage space allocation on HPSS disks. The smallest amount of disk space that can be allocated from a virtual volume is a cluster. The size of the cluster on any given disk volume is determined by the size of the smallest storage segment that will be allocated on the volume, and other factors.
configuration	The process of initializing or modifying various parameters affecting the behavior of an HPSS server or infrastructure service.
COS	Class of Service
Core Server	An HPSS server which manages the namespace and storage for an HPSS system. The Core Server manages the Name Space in which files are defined, the attributes of the files, and the storage media on which the files are stored. The Core Server is the central server of an HPSS system. Each storage sub-system uses exactly one Core Server.
daemon	A UNIX program that runs continuously in the background.
DB2	A relational database system, a product of IBM Corporation, used by HPSS to store and manage HPSS system metadata.
debug	A log record message type used to log lower-level error conditions.
DEC	Digital Equipment Corporation.
delog	The process of extraction, formatting, and outputting HPSS central log records.
deregistration	The process of disabling notification to SSM for a particular attribute change.
descriptive name	A human-readable name for an HPSS server.

device	A physical piece of hardware, usually associated with a drive, that is capable of reading or writing data.
directory	An HPSS object that can contain files, symbolic links, hard links, and other directories.
dismount	An operation in which a cartridge is either physically or logically removed from a device, rendering it unreadable and unwritable. In the case of tape cartridges, a dismount operation is a physical operation. In the case of a fixed disk unit, a dismount is a logical operation.
DNS	Domain Name Service
DOE	Department of Energy
drive	A physical piece of hardware capable of reading and/or writing mounted cartridges. The terms device and drive are often used interchangeably.
EFS	External File System
ERA	Extended Registry Attribute
ESCON	Enterprise System Connection
event	A log record message type used to log informational messages (e.g., subsystem starting, subsystem terminating).
export	\nAn operation in which a cartridge and its associated storage space are removed from the HPSS system Physical Volume Library. It may or may not include an eject, which is the removal of the cartridge from its Physical Volume Repository.
FDDI	Fiber Distributed Data Interface
file	An object than can be written to, read from, or both, with attributes including access permissions and type, as defined by POSIX (P1003.1-1990). HPSS supports only regular files.
file family	An attribute of an HPSS file that is used to group a set of files on a common set of tape virtual volumes.
fileset	A collection of related files that are organized into a single easily managed unit. A fileset is a disjoint directory tree that can be mounted in some other directory tree to make it accessible to users.
fileset ID	A 64-bit number that uniquely identifies a fileset.
fileset name	A name that uniquely identifies a fileset.
file system ID	A 32-bit number that uniquely identifies an aggregate.
FTP	File Transfer Protocol
Gatekeeper	An HPSS server that provides two main services: the ability to schedule the use of HPSS resources referred to as the Gatekeeping Service, and the ability to validate user accounts referred to as the Account Validation Service.

Gatekeeping Service	A registered interface in the Gatekeeper that provides a site the mechanism to create local policy on how to throttle or deny create, open and stage requests and which of these request types to monitor.
Gatekeeping Site Interface	The APIs of the gatekeeping site policy code.
Gatekeeping Site Policy	The gatekeeping shared library code written by the site to monitor and throttle create, open, and/or stage requests.
GB	Gigabyte (2^{30})
GECOS	The comment field in a UNIX password entry that can contain general information about a user, such as office or phone number.
GID	Group Identifier
GK	Gatekeeper
GSS	Generic Security Service
GUI	Graphical User Interface
HA	High Availability
HACMP	High Availability Clustered Multi-Processing - A software package used to implement high availability systems.
halt	A forced shutdown of an HPSS server.
HDM	Shorthand for HPSS/DMAP.
hierarchy	See Storage Hierarchy.
HIMF	HPSS Interim Metadata Format
HiPPI	High Performance Parallel Interface
HPSS	High Performance Storage System
HPSS-only fileset	An HPSS fileset that is not linked to an external filesystem (e.g., XFS).
IBM	International Business Machines Corporation
ID	Identifier
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
Imex	Import/Export
import	An operation in which a cartridge and its associated storage space are made available to the HPSS system. An import requires that the cartridge has been physically introduced into a Physical Volume Repository (injected). Importing the cartridge makes it known to the Physical Volume Library.

I/O	Input/Output
IOD/IOR	Input/Output Descriptor / Input/Output Reply. Structures used to send control information about data movement requests in HPSS and about the success or failure of the requests.
IP	Internet Protocol
IRIX	SGI's implementation of UNIX
junction	A mount point for an HPSS fileset.
KB	Kilobyte (210)
LAN	Local Area Network
LANL	Los Alamos National Laboratory
LARC	Langley Research Center
latency	For tape media, the average time in seconds between the start of a read or write request and the time when the drive actually begins reading or writing the tape.
LCU	Library Control Unit
LDAP	Lightweight Directory Access Protocol
LLNL	Lawrence Livermore National Laboratory
LMCP	Library Manager Control Point
LMU	Library Management Unit
local log	An optional circular log maintained by a Log Client. The central log contains formatted messages from all enabled HPSS servers residing on the same node as the Log Client.
Location Server	An HPSS server that is used to help clients locate the appropriate Core Server and/or other HPSS server to use for a particular request.
Log Client	An HPSS server executing on each HPSS node that is responsible for sending log messages to the local log, to the Log Daemon for central logging, and to SSM to display messages in the Alarm and Event window.
Log Daemon	An HPSS server responsible for writing log messages to the central log.
log record	A record received and maintained in a central log by the HPSS Log Daemon.
log record type	An indicator of whether a message to be logged is an alarm, event, status, debug, request, security, or accounting record.
logging service	An HPSS infrastructure service consisting of a central Log Daemon, one or more Log Clients, and server-specific logging policies.
LRU	Least Recently Used

LS	Location Server
LTO	Linear Tape-Open. A half-inch open tape technology developed by IBM, HP and Seagate.
MAC	Mandatory Access Control
managed object	A programming data structure that represents an HPSS system resource. The resource can be monitored and controlled by operations on the managed object. Managed objects in HPSS are used to represent servers, drives, storage media, jobs, and other resources.
MB	Megabyte (220)
metadata	Control information about the data stored under HPSS, such as location, access times, permissions, and storage policies. Most HPSS metadata is stored in a DB2 relational database.
method	A Java function or subroutine
migrate	To copy file data from a level in the file's hierarchy onto the next lower level in the hierarchy.
Migration/Purge Server	An HPSS server responsible for supervising the placement of data in the storage hierarchies based upon site-defined migration and purge policies.
MM	Metadata Manager. A software library that provides a programming API to interface HPSS servers with the DB2 programming environment.
mount	An operation in which a cartridge is either physically or logically made readable and/or writable on a drive. In the case of tape cartridges, a mount operation is a physical operation. In the case of a fixed disk unit, a mount is a logical operation.
mount point	A place where a fileset is mounted in the XFS and/or HPSS namespaces.
Mover	An HPSS server that provides control of storage devices and data transfers within HPSS.
MPS	Migration/Purge Server
MRA	Media Recovery Archive
MSSRM	Mass Storage System Reference Model
MVR	Mover
NASA	National Aeronautics and Space Administration
Name Service	The portion of the Core Server that provides a mapping between names and machine oriented identifiers. In addition, the Name Service performs access verification and provides the Portable Operating System Interface (POSIX).
name space	The set of name-object pairs managed by the HPSS Core Server.
NERSC	National Energy Research Supercomputer Center
NLS	National Language Support

notification	A notice from one server to another about a noteworthy occurrence. HPSS notifications include notices sent from other servers to SSM of changes in managed object attributes, changes in tape mount information, and log messages that are alarm, event, and status log record message types.
NS	HPSS Name Service
NSL	National Storage Laboratory
object	See Managed Object
ONC	Open Network Computing
ORNL	Oak Ridge National Laboratory
OSF	Open Software Foundation
OS/2	Operating System (multi-tasking, single user) used on the AMU controller PC
PB	Petabyte (2 ⁵⁰)
PFTP	Parallel File Transfer Protocol
physical volume	An HPSS object managed jointly by the Core Server and the Physical Volume Library that represents the portion of a virtual volume. A virtual volume may be composed of one or more physical volumes, but a physical volume may contain data from no more than one virtual volume.
Physical Volume Library	An HPSS server that manages mounts and dismounts of HPSS physical volumes.
Physical Volume Repository	An HPSS server that manages the robotic agent responsible for mounting and dismounting cartridges or interfaces with the human agent responsible for mounting and dismounting cartridges.
PIO	Parallel I/O
PIOFS	Parallel I/O File System
POSIX	Portable Operating System Interface (for computer environments)
purge	Deletion of file data from a level in the file's hierarchy after the data has been duplicated at lower levels in the hierarchy and is no longer needed at the deletion level.
purge lock	A lock applied to a bitfile which prohibits the bitfile from being purged.
PV	Physical Volume
PVL	Physical Volume Library
PVM	Physical Volume Manager
PVR	Physical Volume Repository
RAID	Redundant Array of Independent Disks
RAIT	Redundant Array of Independent Tapes

RAM	Random Access Memory
reclaim	The act of making empty tape virtual volumes available for reuse. Reclaimed tape virtual volumes are assigned a new Virtual Volume ID, but retain the rest of their previous characteristics.
registration	The process by which SSM requests notification of changes to specified attributes of a managed object.
reinitialization	An HPSS SSM administrative operation that directs an HPSS server to reread its latest configuration information, and to change its operating parameters to match that configuration, without going through a server shutdown and restart.
repack	The act of moving data from a virtual volume onto another virtual volume with the same characteristics with the intention of removing all data from the source virtual volume.
request	A log record message type used to log some action being performed by an HPSS server on behalf of a client.
RISC	Reduced Instruction Set Computer/Cycles
RMS	Removable Media Service
RPC	Remote Procedure Call
SCSI	Small Computer Systems Interface
security	A log record message type used to log security related events (e.g., authorization failures).
SGI	Silicon Graphics
shelf tape	A cartridge which has been physically removed from a tape library but whose file metadata still resides in HPSS.
shutdown	An HPSS SSM administrative operation that causes a server to stop its execution gracefully.
sink	The set of destinations to which data is sent during a data transfer (e.g., disk devices, memory buffers, network addresses).
SMC	SCSI Medium Changer
SMIT	System Management Interface Tool
SNL	Sandia National Laboratories
SOID	Storage Object ID. An internal HPSS storage object identifier that uniquely identifies a storage resource. The SOID contains a unique identifier for the object, and a unique identifier for the server that manages the object.
source	The set of origins from which data is received during a data transfer (e.g., disk devices, memory buffers, network addresses).
SP	Scalable Processor

SS	HPSS Storage Service
SSA	Serial Storage Architecture
SSM	Storage System Management
SSM session	The environment in which an SSM user interacts with the SSM System Manager to monitor and control HPSS. This environment may be the graphical user interface provided by the <code>hpssgui</code> program, or the command line user interface provided by the <code>hpssadm</code> program.
stage	To copy file data from a level in the file's hierarchy onto the top level in the hierarchy.
start-up	An HPSS SSM administrative operation that causes a server to begin execution.
status	A log record message type used to log processing results. This message type is being used to report status from the HPSS Accounting process.
STK	Storage Technology Corporation
storage class	An HPSS object used to group storage media together to provide storage for HPSS data with specific characteristics. The characteristics are both physical and logical.
storage hierarchy	An ordered collection of storage classes. The hierarchy consists of a fixed number of storage levels numbered from level 1 to the number of levels in the hierarchy, with the maximum level being limited to 5 by HPSS. Each level is associated with a specific storage class. Migration and stage commands result in data being copied between different storage levels in the hierarchy. Each Class of Service has an associated hierarchy.
storage level	The relative position of a single storage class in a storage hierarchy. For example, if a storage class is at the top of a hierarchy, the storage level is 1.
storage map	An HPSS object managed by the Core Server to keep track of allocated storage space.
storage segment	An HPSS object managed by the Core Server to provide abstract storage for a bitfile or parts of a bitfile.
Storage Service	The portion of the Core Server which provides control over a hierarchy of virtual and physical storage resources.
storage subsystem	A portion of the HPSS namespace that is managed by an independent Core Server and (optionally) Migration/Purge Server.
Storage System Management	An HPSS component that provides monitoring and control of HPSS via a windowed operator interface or command line interface.
stripe length	The number of bytes that must be written to span all the physical storage media (physical volumes) that are grouped together to form the logical storage media (virtual volume). The stripe length equals the virtual volume block size multiplied by the number of physical volumes in the stripe group (i.e., stripe width).

stripe length	The number of bytes that must be written to span all the physical storage media (physical volumes) that are grouped together to form the logical storage media (virtual volume). The stripe length equals the virtual volume block size multiplied by the number of physical volumes in the stripe group (i.e., stripe width).
stripe width	The number of physical volumes grouped together to represent a virtual volume.
System Manager	The Storage System Management (SSM) server. It communicates with all other HPSS components requiring monitoring or control. It also communicates with the SSM graphical user interface (hpssgui) and command line interface (hpssadm).
TB	Terabyte (2 ⁴⁰)
TCP/IP	Transmission Control Protocol/Internet Protocol
trace	A log record message type used to record entry/exit processing paths through HPSS server software.
transaction	A programming construct that enables multiple data operations to possess the following properties: All operations commit or abort/roll-back together such that they form a single unit of work. All data modified as part of the same transaction are guaranteed to maintain a consistent state whether the transaction is aborted or committed. Data modified from one transaction are isolated from other transactions until the transaction is either committed or aborted. Once the transaction commits, all changes to data are guaranteed to be permanent.
TTY	Teletypewriter
UDP	User Datagram Protocol
UID	User Identifier
UPC	Universal Product Code
UUID	Universal Unique Identifier
virtual volume	An HPSS object managed by the Core Server that is used to represent logical media. A virtual volume is made up of a group of physical storage media (a stripe group of physical volumes).
virtual volume block size	The size of the block of data bytes that is written to each physical volume of a striped virtual volume before switching to the next physical volume.
VV	Virtual Volume
XDSM	The Open Group's Data Storage Management standard. It defines APIs that use events to notify Data Management applications about operations on files.
XFS	A file system created by SGI available as open source for the Linux operating system.

Appendix B. References

4. *3580 Ultrium Tape Drive Setup, Operator and Service Guide* GA32-0415-00
5. *3584 UltraScalable Tape Library Planning and Operator Guide* GA32-0408-01
6. *3584 UltraScalable Tape Library SCSI Reference* WB1108-00
7. *AIX Performance Tuning Guide*
8. *Data Storage Management (XDSM) API*, ISBN 1-85912-190-X
9. *HACMP for AIX, Version 4.4: Concepts and Facilities*
10. *HACMP for AIX, Version 4.4: Planning Guide*
11. *HACMP for AIX, Version 4.4: Installation Guide*
12. *HACMP for AIX, Version 4.4: Administration Guide*
13. *HACMP for AIX, Version 4.4: Troubleshooting Guide*
14. *HPSS Error Messages Reference Manual*, July 2008, Release 6.2.
15. *HPSS Programmer's Reference*, July 2008, Release 6.2.
16. *HPSS Programmer's Reference – I/O Supplement*, July 2008, Release 6.2.
17. *HPSS User's Guide*, July 2008, Release 6.2.
18. *IBM 3494 Tape Library Dataserver Operator's Guide*, GA32-0280-02
19. *IBM AIX Version 4.3 Installation Guide*, SC23-4112-01
20. *IBM SCSI Device Drivers: Installation and User's Guide*, GC35-0154-01
21. *IBM Ultrium Device Drivers Installation and User's Guide* GA32-0430-00.1
22. *IBM Ultrium Device Drivers Programming Reference* WB1304-01
23. *Interfacing Guide DAS*, Order no. DOC F00 011
24. *Installing, Managing, and Using the IBM AIX Parallel I/O File System*, SH34-6065-02
25. *Parallel and ESCON Channel Tape Attachment/6000 Installation and User's Guide*, GA32-0311-02
26. *Platform Notes: The hme FastEthernet Device Driver* 805-4449
27. *POSIX 1003.1-1990 Tar Standard*
28. *Reference Guide AMU*, Order no. DOC E00 005
29. *STK Automated Cartridge System Library Software (ACSLs) System Administrator's Guide*, PN 16716
30. *STK Automated Cartridge System Library Software Programmer's Guide*, PN 16718
31. J. Steiner, C. Neuman, and J. Schiller, "*Kerberos: An Authentication Service for Open Network Systems*," USENIX 1988 Winter Conference Proceedings (1988).
32. R.W. Watson and R.A. Coyne, "*The Parallel I/O Architecture of the High-Performance*

- Storage System (HPSS)*,” from the 1995 IEEE MSS Symposium, courtesy of the IEEE Computer Society Press.
33. T.W. Tyler and D.S. Fisher, “*Using Distributed OLTP Technology in a High-Performance Storage System*,” from the 1995 IEEE MSS Symposium, courtesy of the IEEE Computer Society Press.
 34. J.K. Deutsch and M.R. Gary, “*Physical Volume Library Deadlock Avoidance in a Striped Media Environment*,” from the 1995 IEEE MSS Symposium, courtesy of the IEEE Computer Society Press.
 35. R. Grossman, X. Qin, W. Xu, H. Hulen, and T. Tyler, “*An Architecture for a Scalable, High-Performance Digital Library*,” from the 1995 IEEE MSS Symposium, courtesy of the IEEE Computer Society Press.
 36. S. Louis and R.D. Burris, “*Management Issues for High-Performance Storage Systems*,” from the 1995 IEEE MSS Symposium, courtesy of the IEEE Computer Society Press.
 37. D. Fisher, J. Sobolewski, and T. Tyler, “*Distributed Metadata Management in the High Performance Storage System*,” from the 1st IEEE Metadata Conference, April 16-18, 1996.

Appendix C. Developer Acknowledgments

HPSS is a product of a government-industry collaboration. The project approach is based on the premise that no single company, government laboratory, or research organization has the ability to confront all of the system-level issues that must be resolved for significant advancement in high-performance storage system technology.

HPSS development was performed jointly by IBM Worldwide Government Industry, Lawrence Berkeley National Laboratory, Lawrence Livermore National Laboratory, Los Alamos National Laboratory, NASA Langley Research Center, Oak Ridge National Laboratory, and Sandia National Laboratories.

We would like to acknowledge Argonne National Laboratory, the National Center for Atmospheric Research, and Pacific Northwest Laboratory for their help with initial requirements reviews.

We also wish to acknowledge Cornell Information Technologies of Cornell University for providing assistance with naming service and transaction management evaluations and for joint developments of the Name Service.

We also wish to acknowledge the many discussions, design ideas, implementation and operation experiences we have shared with colleagues at the National Storage Laboratory, the IEEE Mass Storage Systems and Technology Technical Committee, the IEEE Storage System Standards Working Group, and the storage community at large.

We also wish to acknowledge the Cornell Theory Center and the Maui High Performance Computer Center for providing a test bed for the initial HPSS release.

Finally, we wish to acknowledge CEA-DAM (**Commissariat à l'énergie Atomique - Centre d'études Bruyres-le-Château**) for providing assistance with development of NFS V3 protocol support.

Appendix D. HPSS.conf Configuration File

The **HPSS.conf** configuration file contains tuning options to be used by HPSS clients and servers. For additional information, please see the **HPSS.conf** manual page.

General HPSS.conf Rules/Suggestions:

- Keywords MUST be specified precisely as shown (no extra spaces)
- Lines are comprised of *comments*, *blank lines*, *simple specifiers*, *specifiers* and *values* - “abc = def”, or *compound specifiers* and *terminators* - “def = { ...}.”
- A “;” is used to comment out (deactivate) an actual Configuration Option. To activate these options, remove the “;” (Suggestion)
- Statements with a “#” sign as the first non-white character are explanatory comments. (Suggestion)
- Only ten levels of specification are allowed: Stanzas, SubStanzas, Sections, and SubSections.
- SubStanzas may only exist in Compound Stanzas. Sections may only exist in Compound SubStanzas, and SubSections may only exist in Compound Sections.
- No line may exceed 512 characters, including the “= {“.
- Comments may be included by entering either a semicolon (“;”) or a pound sign (“#”) as the first non-white character in a line. Use of either of these characters other than as the first character will not be interpreted as a comment (It will be interpreted as part of the specifier or value!). Suggestion - Do NOT put comments at the end of the line!
- Indentation is optional but is strongly encouraged (assists in diagnosis). Use “tabs” for indentation. (Suggestion)
- Closing braces (“}“) must be used to terminate opening braces (“ = {“). This MUST appear on a separate line. PLEASE maintain indentation.
- Spaces before or after the equal sign (“=”) are optional.
- Blank Lines are ignored.
- The Non-HPSS Parallel FTP Daemon options should be left alone. (Suggestion)



NOTE: *HPSS and Network Tuning are highly dependent on the application environment. The values specified herein are **NOT** expected to be applicable to any installation!*

D.1. PFTP Client Stanza

The Parallel FTP Client configuration options are in two distinct stanzas of the **HPSS.conf** file (PFTP Client Stanza and PFTP Client Interfaces Stanza).

Table 12. PFTP Client Stanza Fields

Configuration Type	Abbreviated Description
--------------------	-------------------------

Stanza (CMPD)	<p>PFTP Client = {</p> <p>E.g. PFTP Client = {</p> <p><i>Optional Reserved Stanza specifier.</i></p> <p>Must be terminated with a matching “}”</p>
SubStanza	<p>SYSLOG Facility = <value></p> <p>Values: DAEMON, LOCAL0 ... LOCAL7</p> <p>E.g. SYSLOG Facility = LOCAL2</p> <p><i>Optional SubStanza specifying the Syslog Facility for the MultiNode Daemon</i></p>
SubStanza	<p>Debug Value = <value></p> <p>Values: 0 – 3</p> <p>E.g. Debug Value = 1</p> <p><i>Optional SubStanza specifying the Level of Debugging for the Parallel FTP Client. Larger number provides more information.</i></p>
SubStanza	<p>Authentication Mechanism = <value></p> <p>Values: USER_PASS (Default), GSS, IDENT</p> <p>E.g. Authentication Mechanism = GSS</p> <p><i>Optional SubStanza specifying the preferred Client Authentication mechanism. NOTE: this should be GSS if Kerberos Credentials are to be used.</i></p>
SubStanza	<p>Default COS = <value></p> <p>E.g. Default COS = 99</p> <p><i>Optional SubStanza specifying the default Class of Service if not explicitly specified by the user. Use with caution – standard procedure is to allow the Core Service to determine the optimal COS!</i></p>
SubStanza	<p>Protocol = <value></p> <p>Values: PDATA_AND_MOVER (Default), PDATA_ONLY, PDATA_PUSH</p> <p>E.g. Protocol = PDATA_ONLY</p> <p><i>Optional SubStanza specifying the default protocol. May contain any of the three protocols supported.</i></p>

SubStanza	<p>Auto Parallel Size = <value></p> <p>Value: Size - May be specified as a decimal number or “xMB” style notation.</p> <p>E.g. Auto Parallel Size = 4MB</p> <p><i>Optional</i> SubStanza specifying the minimum file size to start using the “auto-parallel” features of the PFTP Client.</p>
SubStanza	<p>PortRange = <value></p> <p>Value: ncadg_ip_tcp[StartPort-EndPort]</p> <p>E.g. PortRange = ncadg_ip_tcp[10100-12100]</p> <p><i>Optional</i> SubStanza specifying the TCP port range to use between the PFTP Client and the Mover(s). This may be necessary when Port Range Filters are used for Security.</p>
SubStanza	<p>Parallel Block Size = <value></p> <p>Value: Size - May be specified as a decimal number or “xMB” style notation.</p> <p>E.g. Parallel Block Size = 512KB</p> <p><i>Optional</i> SubStanza specifying the size of the data blocks to be used for parallel data transfers.</p>
SubStanza	<p>Transfer Buffer Size = <value></p> <p>Value: Size - May be specified as a decimal number or “xMB” style notation.</p> <p>E.g. Transfer Buffer Size = 16MB</p> <p><i>Optional</i> SubStanza specifying the PFTP Buffer sizes.</p>
SubStanza	<p>Socket Buffer Size = <value></p> <p>Value: Viable Socket Size - May be specified as a decimal number or “xMB” style notation.</p> <p>E.g. Socket Buffer Size = 16MB</p> <p><i>Optional</i> SubStanza specifying the Pdata Socket Buffer sizes.</p>
SubStanza	<p>MAX Ptran Size = <value></p> <p>Value: Size - May be specified as a decimal number or “xMB” style notation.</p> <p>E.g. MAX Ptran Size = 4GB</p> <p><i>Optional</i> SubStanza specifying a larger transfer size between socket open and closure. For disk COSSs, the segment sizes may potentially override this specification!</p>

SubStanza	No SAN3P E.g. No SAN3P <i>Optional</i> SubStanza specifying not to use SAN3P even if it is available. Default is to use SAN3P if available.
SubStanza	No Transfer Agent Support E.g. No Transfer Agent Support <i>Optional</i> SubStanza specifying to NOT use the Transfer Agent. Default is to use the Transfer Agent.
SubStanza	No 64-bit Support E.g. No 64-bit Support <i>Optional</i> SubStanza specifying to NOT use the 64-bit protocol. Default is to use the 64-bit protocol.
SubStanza	Special Features Enabled E.g. Special Features Enabled <i>Optional</i> SubStanza for Performance Testing ONLY. Should NOT be active except by appropriate personnel. Default is Off.

Note: All PFTP Client SubStanzas are optional.

The **PFTP Client** = { ... } stanza contains several optional specifications for the **pftp_client** executables.

The **SYSLOG Facility** = **value** is used to establish the `syslog` facility for the Multinode Daemon. It has no relevance to the PFTP Client running without the Multinode Daemon.

The **Debug Value** = **value** is used to provide additional diagnostic information for the PFTP Client. Under normal circumstances, this should be set to 0 (Default) or 1. Larger values will provide additional information; but, will cause users to complain!

The **Authentication Mechanism** = **value** is used to determine the preferred authentication mechanism for the PFTP Client. If the desired mechanism is to use Kerberos Credentials, this should be activated and set to **GSS**. Unless this is appropriately set, the PFTP Client will NOT use Kerberos Credentials even if they exist. The final determination of permitted mechanisms is performed by the PFTP Server.

The **DEFAULT COS** = **value** sub stanza assigns a default Class of Service (COS) for HPSS. If this option is not specified, the Core Server (Bitfile component) is responsible for determining the default COS unless the user explicitly issues a “`site setcos <ID>`” command or specifies “`-C<ID>`” on the command line to change the class of service. If this sub stanza is specified, the class of service provided in **value** will be used unless the user explicitly issues a “`site setcos <ID>`” command to change the class of service. For this reason, caution should be used in specifying this option.

The **Protocol** = **value** sub stanza is used to specify the desired PFTP protocol. Currently, any of three values may be specified: **PDATA_AND_MOVER**, **PDATA_ONLY**, or **PDATA_PUSH**. The default specification is **PDATA_AND_MOVER**. The **PDATA_ONLY** specification provides improved performance in high latency WAN environments.

The `pftp_client` automatically performs conversion of **get** and **put** commands to their parallel equivalents, **pget** and **pput**. Some sites have reported degraded performance as a result of this substitution occurring with small file transfers. To accommodate this problem, the **Auto Parallel Size = value** sub stanza may be specified in the **HPSS.conf** file where the “automatic” parallel features will **NOT** be invoked if the size of the file is less than the value provided. The value may be specified as a decimal number (1048576) or in the format: `xMB`.

For sites with “Firewalls/ Diodes” installed, it may be necessary to provide specific ports for the data to move between the PFTP Client and the Mover(s). This can be accomplished by specifying the **PortRange = value** sub stanza. The syntax of this statement is:

```
PortRange = ncadg_ip_udp[10100-12100]:ncadg_ip_tcp[10100-12100]
```

This syntax of the value is identical to DCE’s **RPC_RESTRICTED_PORTS** environment variable. Only the **ncacn_ip_tcp[start_port-end_port]** (*TCP* component) is used so the **ncadg_ip_udp** component may be omitted. At this time, the restricted ports are ignored for passive transfers. Arbitrary ports will be assigned!

Additional options are available for controlling the size of the PFTP transfer buffers, **Transfer Buffer Size**, and the buffer size for the sockets in the **PDATA_ONLY** protocol, **Socket Buffer Size**. The value may be specified as a decimal number (1048576) or in the format: `xMB`.

The PFTP parallel protocol opens and closes the sockets between the PFTP Client child processes and the Mover(s). The default value for tape was every 512 MB and for disk was the smaller of the size of 64 storage segments or 512 MB. With transfers increasing in performance into the 100MB/sec and greater range, the opening and closing of sockets is another performance problem. The **MAX Ptran Size = value** sub stanza has been provided to allow for larger transfers between socket open and closing.

NOTE: in the case of disks, the 64 storage segments is still the overriding specification, so Storage Classes need careful specification to provide very large segments if the value associated with **MAX Ptran Size** is large. An artificial limit of 250 GB is compiled into the PFTP Client, which should not cause a great concern any time in the near future. Even at 1 GB/sec, this is several minutes! The value may be specified as a decimal number (4294967296) or in the format: `xGB`.

Under normal operating conditions, the **No SAN3P**, **No Transfer Agent Support**, and **Special Features Enabled** components should remain commented out.

PFTP Client Stanza Example (with suggested contents):

```
PFTP Client = {
  # SYSLOG facility to use for all syslog messages by the Multinode
  Daemon.
  SYSLOG Facility = LOCAL0
  # Set Default Class of Service
  ; DEFAULT COS = 2
  # Debugging Levels
  ; Debug Value = 1
  # Set Default Protocol
  ; Protocol = PDATA_AND_MOVER
  # Set Minimum File Size for auto mapping Non-parallel
  # commands to Parallel commands
  Auto Parallel Size = 4MB
  # Set the Port Range for Port Restrictions on Parallel Transfers
  # The syntax has been taken from DCE.
  ; PortRange = ncadg_ip_udp[10100-12100]:ncadg_ip_tcp[10100-12100]
  # PDATA Options
  ; Parallel Block Size = 512KB
```

```

; Transfer Buffer Size = 1MB
; Socket Buffer Size = 16MB
# PFTP sets an Artificial (Compiled in) Maximum of 250GB
MAX Ptran Size = 10GB
# Disable SAN3P
; No SAN3P
# Disable Transfer Agent
; No Transfer Agent Supported
# Disable 64-bit Protocol - Default is on
; No 64-bit Support
# Special Features
; Special Features Enabled
}

```

D.2. PFTP Client Interfaces Stanza

Many systems have multiple interfaces, some of which may not have access to all destination hosts. The PFTP Client Interfaces stanza is used to specify which interfaces on the source host should be used to communicate to destination Parallel FTP Daemons and/or HPSS Movers. This is particularly useful if both low speed and high speed interfaces are available to the client host and the PFTP data transfers should use the high speed interfaces.

Table 13. PFTP Client Interfaces Stanza Fields

Configuration Type	Abbreviated Description
Stanza (CMPD)	PFTP Client Interfaces = { E.g. PFTP Client Interfaces = { <i>Optional Reserved</i> Stanza specifier. Must be terminated with a matching “}”
SubStanza (CMPD)	<Hostname> <hostname.domain> = { E.g. my_host my_host.my.domain = { Contains the hostname(s) executing the PFTP Client. Must be terminated with a matching “}” Multiple Hostname Substanzas may be in a single HPSS.conf file representing multiple PFTP Client hosts sharing the HPSS.conf file.
Section (CMPD)	<Name> <Name> = { Name: One or more hostnames E.g. storage storage.my.domain = { Contains the hostname(s) executing the PFTP Daemon. Must be terminated with a matching “}” Multiple Daemon Sections may be in a single Hostname Substanza representing multiple PFTP Daemon destinations which may use different characteristics.

SubSection	<Name> or <Dotted IP Address> Name: Valid Interface Name Dotted IP Address: 132.175.1.1 E.g. eth0 132.175.1.1 <i>Optional</i> parameter containing the name or Dot Notation IP Address specification for the interface on the local host (PFTP Client) to use to connect to the Mover(s) associated with the specified PFTP Daemon.
------------	---

The **PFTP Client Interfaces = { ... }** stanza contains several configuration options for the **pftp_client** executables.

SubStanzas refer to the **hostname(s)** associated with the local system where the pftp_client is being invoked.

Sections refer to the **Parallel FTP Daemon hostname(s)** where the PFTP Daemon is being invoked.

SubSections refer to the Network Interface to be utilized (by the host where the PFTP client is invoked) to transfer data to the HPSS Mover systems.



*For HPSS, this is **not** necessarily the name of the Mover(s).*

SubSections specify names or Dot Notation IP Addresses of the interfaces on the local host to be used. For HPSS, all of these interfaces must be able to connect to the Mover(s). NOTE: If and only if a specific COS is specified, these interfaces need only provide connection to the Mover(s) associated with the specific COS.

PFTP Client Interfaces Stanza Rules:

- Source hostnames may contain one or more hostnames separated by white spaces (subject to the HPSS.conf line character limit).
- "Default" is a reserved notation to be used if the local host is not in the Stanza.
- Destination Host (FTP Daemon Host) may contain one or more hostnames separated by white spaces (subject to the HPSS.conf line character limit).
- Interface Specification must be specified by interface name or IP Address Dot Notation.
- Interfaces **must** be able to connect to destination (HPSS Mover.)
-



Communication failures that are not easily diagnosed will occur if the interface specification is invalid.

The following example is completely commented out. The default interface(s) will be used. This is probably typical for many sites and does not need to be modified unless multiple interfaces and asymmetric networks are involved.

PFTP Client Interfaces Stanza Example:

```
; PFTP Client Interfaces = {
  # PFTP Client Host Name(s)
  ; water.clearlake.ibm.com water = {
    # Next Specification is the PFTP Daemon Host
    # water has 3 specific interfaces that can talk
    # to the HPSS Movers associated with the PFTP
    # Daemon Host "water", as well as various
    # interfaces of the form 192.2.nnn.nnn
    ; water.clearlake.ibm.com water = {
      # Interfaces on the host specified as the Client Machine
      ; 192.94.47.227
      ; 192.175.14.35
      ; 192.222.197.1
      ; eth*
    ; }
    # water has ONLY 1 interface that can talk to the HPSS
    # Movers associated with the PFTP Daemon Host "sneezy"
    ; sneezy sneezy.clearlake.ibm.com = {
      ; 192.94.47.227
    ; }
    # Use the default water interface, 192.100.101.1, to talk
    # to any other PFTP Daemons.
    ; Default = {
      ; 192.100.101.1
    ; }
  ; }

; sneezy sneezy.clearlake.ibm.com = {
  ; larry larry.clearlake.ibm.com = {
    ; 192.94.47.226
  ; }
  ; sneezy sneezy.clearlake.ibm.com = {
    ; 192.94.47.226
  ; }
; }

# For all other Client Hosts - This allows a single HPSS.conf
# file to be available using a common files system. This is
# ONLY useful for cluster systems that specify "Common"
# interfaces for multiple # nodes of the cluster (I/O Nodes)
; Default = {
  # Client Host Name
  ; water water.clearlake.ibm.com = {
    ; 134.253.14.227
  ; }
; }
; }
```

D.3. Multinode Table Stanza

The HPSS PFTP Client normally forks children to provide multiple network paths between the PFTP Client and the Mover(s). In some instances, it may be preferable to have these processes (pseudo children) running on independent nodes. In this case, it is necessary to setup a **multinoded** daemon on the independent node/host and have the PFTP client initiate the data transfer process(es) with these child processes. The **Multinode Table** stanza is used to specify what remote hosts are to perform the “pseudo” PFTP Client child processes functions.

Table 14. Multinode Table Stanza Fields

Configuration Type	Description
Stanza (CMPD)	Multinode Table = { E.g. Multinode Table = { <i>Optional Reserved</i> Stanza specifier. Must be terminated with a matching “}”
SubStanza	Sleep for Debugger = values Value: Time in seconds. E.g. Sleep for Debugger = 15 <i>Optional</i> parameter to specify a delay in the Multinode Daemons to allow diagnosis. This should ONLY be specified for diagnostics and will unnecessarily cause degradation if misused! Leave commented out.
SubStanza (CMPD)	<Local Hostname(s)> E.g. my_host my_host.my.domain = { Contains the local hostname(s) this SubStanza represents. MUST be terminated with a matching “}”
Section (CMPD)	<Remote Hostname(s)> E.g. FTP_host PFTP_host.domain = { Contains the hostname(s) of the system running the PFTP Server. MUST be terminated with a matching “}”
Sub-Section	<remote_host> or <remote_host> = <Dot Notation Interface> E.g. his_name or his_name = 100.102.103.45 Contains the hostname in either string format or Dot Notation IP Address of the host to act as a “Pseudo” PFTP Child. If a secondary name is specified after the “=”, the first interface is to be used for the “control” connection between the PFTP Client and the Multinoded hosts and the second specification is the interface to be used for the “data” connection(s) to the Mover(s). If only one value is provided, it represents BOTH the “control” and “data” connections.

The **Multinode Table** = { ... } stanza contains one or more substanzas specifying the names of the host initiating the PFTP session.

Each section contains one or more names/IP addresses of remote hosts executing a Multinode Daemon (**multinoded**). The remote host must have appropriate entries for the **inetd** or **xinetd**

superdaemon (**/etc/inetd.conf** and **/etc/services**) to initiate the multinoded.

The sections may be either a simple section or a valued section. A simple stanza is a single name/Dot Notation IP Address to be used for both “Control” connection and “Data” connection. The valued stanza is used to specify the name/Dot Notation IP Address for the “Control” connection (specifier) and the name/Dot Notation IP Address for the “Data” connection (value.)

Multinode Table Stanza Rules:

- **SubStanza** hostnames (local hosts) may contain one or more hostnames separated by white spaces (subject to the HPSS.conf line character limit.)
- **Section** hostnames (remote hosts) [and/or values] may be specified as either string-based hostnames or Dot Notation IP Addresses. Only one entry per line.

Multinode Table Example:

```
# Options read by the Multinode Daemon
Multinode Table = {
  # Diagnostic Delay
  ; Sleep for Debugger = 15
  # Hostname of the Client
  water water.clearlake.ibm.com =
    # Name of the system running the PFTP Server
    pftp_server pftp_server.clearlake.ibm.com = {
      # Specification of the Multinode Host
      # If the Data Node is a different interface than the interface
      # specified by the Control Node; then enter the Data Node
      # Interface after the "=" otherwise the Control and Data
      # are the same.
      # Control and/or Data may be dot notation OR string hostnames.
      Water = 134.253.14.227
    }
  }
  Default = {
    Default = {
      # If the Data Node is different than the Control Node
      # Enter the Data Node after the "=" otherwise the
      # Control and Data are the same.
      # Control and/or Data may be dot notation OR string hostnames.
      larry = sneezy
    }
  }
}
```

D.4. Network Option Stanza

The Network Options are in the **Network Options = { ... }** stanza of the **HPSS.conf** file.

The **Network Options** stanza allows different options to be specified based on: Source IP address [Local Interface Name(s)] AND Destination IP Address(es). When the Parallel FTP Client, Client API, or Mover establish connections, they will search the contents of this file for entries matching Source/Destination IP addresses and use the options specified in the matching entry.

The configuration file entries contain values/flags to be used for applying assorted socket and network options including: whether to enable/disable TCP Delay (**TcpNoDelay**), the socket send sizes (**SendSpace**) and/or socket receive sizes (**RecvSpace**), the desired write buffer size (**WriteSize**), and **RFC1323** support (“Large” Window.)

Which configuration entry to use is determined based on the Local Interface Name and the

Destination IP address “masked” by the **NetMask** value. The calling application (PFTP Client, Client API, or Mover) will apply the value of the NetMask specification in the configuration file entry to the specified destination address. A “**Default**” destination may be specified for all sources/destinations not explicitly specified in the HPSS.conf file.

Table 15. Network Options Stanza Fields

Configuration Type	Description
Stanza (CMPD)	<p>Network Options = {</p> <p>E.g. Network Options = {</p> <p><i>Optional Reserved</i> Stanza specifier.</p> <p>Must be terminated with a matching “}”</p>
SubStanza	<p>Default Write Size = <value></p> <p>E.g. Default Receive = 1MB</p> <p><i>Optional</i> SubStanza specifying the default network Read socket size if not specified explicitly.</p>
SubStanza	<p>Default Write Size = <value></p> <p>E.g. Default Send Size = 1MB</p> <p><i>Optional</i> SubStanza specifying the default network write socket size if not specified explicitly.</p>
SubStanza	<p>Default Write Size = <value></p> <p>E.g. Default Write Size = 4MB</p> <p><i>Optional</i> SubStanza specifying the default write size if not specified explicitly.</p>
SubStanza (CMPD)	<p><Source IP Interface Name> = {</p> <p>E.g. my_host my_host.my.domain = {</p> <p><i>Optional</i> SubStanza specifying an interface name. May contain one or more names separated by white spaces.</p> <p>May contain: “Default = {“ (<i>Reserved</i> Specification) for inclusion of entries not explicitly specified.</p> <p>Must be terminated with a matching “}”</p>

Section (CMPD)	<p><Destination IP Address> = { E.g. 100.101.102.0 = { <i>Optional</i> SubStanza specifying a dotted decimal address of the destination interface. Only one address is allowed; however, networks and sub-networks may be chosen by appropriate specification of the NetMask. May contain: “Default = {“ (<i>Reserved</i> Specification) for inclusion of entries not explicitly specified. Must be terminated with a matching “}”</p>
SubSection	<p>NetMask = <value> Value: Viable netmask as IP Address E.g. NetMask = 255.255.255.0 <i>Optional</i> parameter to specify the dotted decimal net mask to apply to the Destination IP Address to determine whether the entry applies</p>
SubSection	<p>RFC1323 = <value> Values: 0, 1 E.g. RFC1323 = 1 <i>Optional</i> parameter to specify whether the RFC1323 option should be disabled (0) or enabled (any other value)</p>
SubSection	<p>SendSpace = <value> Values: Size specified as decimal value or "xMB" format E.g. SendSpace = 256KB <i>Optional</i> parameter to specify the value to be used for the socket sending buffer space.</p>
SubSection	<p>RecvSpace = <value> Values: Size specified as decimal value or "xMB" format E.g. RecvSpace = 256KB <i>Optional</i> parameter to specify the value to be used for the socket receive buffer space.</p>
SubSection	<p>WriteSize = <value> Values: Size specified as decimal value or "xMB" format E.g. WriteSize = 1MB <i>Optional</i> parameter used to set the size to be used for each individual write request to the network</p>

SubSection	<p>TcpNoDelay = <value></p> <p>Values: 0, 1</p> <p>E.g. TcpNoDelay = 1</p> <p><i>Optional</i> parameter Indicates whether the TCP Delay option should be disabled (0) or enabled (any other value)</p>
------------	---

SendSpace & RecvSpace Controls the size of the receive and send buffers for TCP/IP sockets. Internally, HPSS servers and clients attempt to set these buffers sizes explicitly, but other utilities may not. Typically, the RecvSpace and the SendSpace are equal; however, this is not mandated. Setting either of these values in excess of the system maximum will result in a value <= the system maximum. The maximum can be observed/changed on AIX using the “no” command and observing/setting the **sb_max** parameter. There is no portable mechanism for determining the system maximum. Consequently, the specified values may be reduced until an acceptable value is obtained. This process involves a bit-shift operation (divide by 2.)

RFC1323 Controls whether large TCP window sizes are used. Usually turned on (1) for higher throughput networks (e.g. SP/x switch, Gigabit Ethernet, etc.) and turned off (0) for lower throughput networks (e.g. 10/100 Mb ethernet, FDDI, etc.) Large windows provide for increased performance over some networks, but may have a negative performance impact on others. NOTE: currently the ability to enable or disable **RFC 1323** support in this manner is specific to AIX. (An equivalent setting for Solaris is the **tcp_wscale_always** flag, set with the command “**ndd /dev/tcp tcp_wscale_always**”.)

The **WriteSize** allows the size of the individual write requests to the TCP/IP connections to be configured. The default behavior (if no entry in the file matches a connection or if zero is entered for the value of this field) is that the size of the write request is the size of the data buffer. On some networks (e.g. the SP/x switch), improved performance has been measured by using a smaller value (e.g. 32KB) for the size of the individual writes to the network. If no entry is found that matches a network connection or the value specified is zero, HPSS will query an environment variable, **HPSS_TCP_WRITESIZE**, and use that value, if set and non-zero, for the write size.

The **TcpNoDelay** option determines whether HPSS will enable or disable the algorithm that tries to improve performance from small network writes. This algorithm attempts to coalesce small writes to a TCP/IP connection so they can be sent in a single packet by delaying physical writes to the network. HPSS typically disables this algorithm so that delays are not experienced while sending Mover Protocol and parallel data transfer headers. However, if this causes a performance degradation on a specific network (e.g., causes smaller than optimal packet sizes for large transfers), this can be disabled for data transfer connections.

Network Options Stanza Specific Rules:

- The first matching entry found in the file will be used to determine the network options used for that connection.
- Multiple “Source Interface Name” SubStanzas may be included within the “Network Options” Stanza. A “Default” Source Interface Name SubStanza may be specified.
- The Source Interface Name SubStanza may specify one or more names [subject to the HPSS.conf line character limit (including the “= {“.)] NOTE: Do not include the quotes when specifying Default.
- Destination IP Address must be specified in Decimal Dot Notation.

- Multiple Sections may be included in any SubStanza. A “Default” Destination Interface Name Section may be specified. NOTE: Do not include the quotes when specifying Default.
- The NetMask must be specified in Decimal Dot IP Address Notation
- All SubSections must be specified in every Section.

Network Options Stanza Example:

NOTE: Tuning is a “fine art” and may vary dramatically within any network configuration and may change with only very minor network configuration modifications. Values provided below are not necessarily “good” numbers!

```
# HPSS Network Options
Network Options = {
    ; Default Receive Size = 1MB
    ; Default Send Size = 1MB
    ; Default Write Size = 4MB
    # My Interface specification(s) using Interface Name
    # Notation
    my_host my_host.domain = {
        # Destination IP Address in Dot Notation
        100.101.102.103 = {
            # The netmask to be applied to the Dest. IP Address
            Netmask = 255.255.255.0
            # Use large IP Windows
            RFC1323 = 1
            # Socket Transmission Size.
            SendSpace = 1048576
            # Socket Receive Size.
            RecvSpace = 1MB
            # The overall buffer size to use for writing.
            WriteSize = 2MB
            # The TCP No Delay Flag is disabled
            TCPNoDelay = 0
        }
        # Default Destination - options to be used for destinations
        # NOT explicitly specified.
        Default = {
            NetMask = 255.255.255.0
            RFC1323 = 1
            SendSpace = 512KB
            RecvSpace = 512KB
            WriteSize = 256KB
            TCPNoDelay = 1
        }
    }
}
# Values to be used for source hosts not explicitly specified
Default = {
    # Destination IP Address in Dot Notation
    200.201.202.203 = {
        NetMask = 255.255.255.0
        RFC1323 = 1
        SendSpace = 1048576
        RecvSpace = 1MB
        WriteSize = 2MB
        TCPNoDelay = 0
    }
    # Default Destination - options to be used for destinations
    # NOT explicitly specified.
```



```

    Default = {
        NetMask = 255.255.255.0
        RFC1323 = 1
        SendSpace = 256KB
        RecvSpace = 128KB
        WriteSize = 512KB
        TCPNoDelay = 0
    }
}
}

```

D.5. PFTP Daemon Stanza

A large number of options are available for configuring the PFTP daemon and tuning its performance. These options were previously specified in the ftpaccess file or via command line switches. These options have now been consolidated into the PFTP daemon stanza in the HPSS.conf file. The options are described below:

Table 16. PFTP Daemon Stanza Description

Configuration Type	Abbreviated Description
Stanza (CMPD)	<p>PFTP Daemon = {</p> <p><i>Optional Reserved</i> Stanza specifier on Client ONLY Machines. Required Stanza on all HPSS PFTP Server systems.</p> <p>Must be terminated with a matching “}”</p>
SubStanza	<p>Allow Core Files</p> <p><i>Optional</i> SubStanza specifying that the system should save Core Files if the PFTP Daemon crashes. xinetd disables core files by default.</p>
SubStanza	<p>Core File Directory = <value></p> <p>Value: Pathname</p> <p><i>E.g. Core File Directory = /var/hpss/adm/core/PFTP_Daemon</i></p> <p><i>Optional</i> Substanza to specify the Directory where the PFTP Server should put Core files.</p>

SubStanza	<p>SYSLOG Facility = <value></p> <p>Value: DAEMON, LOCAL0 ... LOCAL7</p> <p><i>E.g. SYSLOG Facility = LOCAL0</i></p> <p>Replaces -s<string> option.</p> <p><i>Optional</i> SubStanza specifying the syslog facility for the HPSS PFTPD. The default syslog facility is LOG_DAEMON (reference: /usr/include/sys/syslog.h). Alternatives are LOCAL0 - LOCAL7. Incorrect specification will default back to LOG_DAEMON. To make use of the alternates, modify /etc/syslog.conf to use the alternate facility. Note, the file specified in the /etc/syslog.conf must exist prior to initialization/refresh of the syslogd.</p>
SubStanza	<p>Use Foreign LDAP for Cross Realm</p> <p><i>Optional</i> SubStanza specifying that LDAP lookups should use the LDAP Server in the Foreign Realm. Few sites want this option.</p>
SubStanza	<p>No Transfer Agent Support</p> <p><i>Optional</i> SubStanza used to disable the Transfer Agent support for the PFTP Daemon.</p>
SubStanza	<p>Use the KDC Registry</p> <p><i>E.g. Use KDC Registry</i></p> <p>Replaces -S option.</p> <p><i>Optional</i> SubStanza specifying use of the Kerberos KDC (registry) for authentication (bypassing the passwd file).</p>
SubStanza	<p>FTP Base Directory = <value></p> <p>Value: Pathname</p> <p><i>E.g. FTP Base Directory = /var/hpss</i></p> <p>Replaces -D<string> option.</p> <p><i>Optional</i> SubStanza setting the {FTPBaseDir} path. Default: /var/hpss. This directory must contain several sub-directories including: adm, bin, daemon, and etc. Specific files/sub-directories are located in each of these subdirectories - etc: ftpaccess, [ftpbanner], ftpusers, and [trusted_hosts]. adm: [daemon.syslog], [hpss_ftp.log], [xferlog]. daemon: ftpd/ftp.pids-hpss_class. [] implies optional others are required. etc/passwd is optional for FTP if Use the KDC Registry or Use Extended Registry Attributes is specified.</p>

SubStanza	<p>FTP Access File = <value></p> <p>Value: filename</p> <p><i>E.g.</i> FTP Access File = myftpaccess</p> <p>Replaces -F<string> option.</p> <p><i>Optional</i> SubStanza setting the {FTP_FtpAccessFile}. Default: ftpaccess. Located in the directory {FTPBaseDir}/etc.</p>
SubStanza	<p>Disable Slash Home Directory</p> <p><i>E.g.</i> Disable Slash Home Directory</p> <p>Replaces -Z option.</p> <p><i>Optional</i> SubStanza disabling use of / as the user's home directory. Normally, this should be active for Security reasons.</p>
SubStanza	<p>Disable Access if no Home Directory</p> <p><i>E.g.</i> Disable Access if No Home Directory</p> <p>Replaces -H option.</p> <p><i>Optional</i> SubStanza disallowing login for users whose home directory does not exist or is not properly configured. The default behavior is to put the user in the "/" directory. Normally, this should be active for Security reasons.</p>
SubStanza	<p>Disable Hints</p> <p><i>E.g.</i> Disable Hints</p> <p>Replaces hpss_option HINTS off in ftpaccess.</p> <p><i>Optional</i> SubStanza disabling the sending of hints to the Core Server. (HPSS only.) Not recommended.</p>
SubStanza	<p>Allow Trusted Hosts Authentication</p> <p><i>E.g.</i> Allow Trusted Hosts Authentication</p> <p>Replaces -I option.</p> <p><i>Optional</i> SubStanza to permit use of the Trusted Hosts File. Note: this is STRONGLY Discouraged. (HPSS only.)</p>
SubStanza	<p>HPSS FTP Principal = <value></p> <p>Value: Appropriate HPSS Principal Name</p> <p><i>E.g.</i> HPSS FTP Principal = hpssftp</p> <p>Replaces -P option and hpss_option PRINC name in ftpaccess.</p> <p><i>Optional</i> SubStanza specifying the HPSS principal representing hpssftp. (HPSS only.)</p>

SubStanza	<p>FTP Principal Keytab File = <value></p> <p>Value: Pathname/Filename</p> <p><i>E.g. FTP Principal Keytab File = /var/hpss/etc/hpss.keytabs</i></p> <p><i>Optional</i> SubStanza specifying the keytab containing the FTP principal.</p>
SubStanza	<p>Allow Passive Connections</p> <p><i>E.g. Allow Passive Connections</i></p> <p>Replaces -A option.</p> <p><i>Optional</i> SubStanza enabling passive connections. Note: Not supported in the HPSS parallel FTP daemon for parallel operations. Client requests to use passive mode will be rejected.</p>
SubStanza	<p>Sleep for Debugger = <value></p> <p><i>E.g. Sleep for Debugger = 5</i></p> <p>Replaces -z option.</p> <p><i>Optional</i> SubStanza specifying the number of seconds for the HPSS PFTP Daemon to sleep at initialization. Useful when attempting to attach to the daemon with the debugger. NOTE: leaving this active will cause significant degradation to the PFTP service!</p>
SubStanza	<p>Must Have Credentials</p> <p><i>E.g. Must Have Credentials</i></p> <p>Replaces -a option.</p> <p><i>Optional</i> SubStanza mandating authentication with Kerberos credentials, disabling {Username}/{Password} authentication. This also disables the user command.</p>
SubStanza	<p>Use Channel Bindings</p> <p><i>E.g. Use Channel Bindings</i></p> <p><i>Optional</i> Kerberos allows for extra security by using channel bindings in credentials. This has been turned off; but can be reset by inclusion of this option.</p>
SubStanza	<p>Special Features Enabled</p> <p><i>E.g. Special Features Enabled</i></p> <p><i>Optional</i> SubStanza for Performance Testing ONLY. Should NOT be active except by appropriate personnel. Default is Off.</p>
SubStanza	<p>Allow CCC Command</p> <p><i>E.g. Allow CCC Command</i></p> <p><i>Optional</i> Kerberos option not relevant to the HPSS PFTP Daemon.</p>

SubStanza	<p>PFTP IO Buffer Size = <value></p> <p><i>E.g. PFTP IO Buffer Size = 4MB</i></p> <p>Replaces -b<string> option.</p> <p><i>Optional</i> SubStanza setting the preferred IO Buffer Size for the PFTP Server</p>
SubStanza	<p>Debug Value = <value></p> <p><i>E.g. Debug Value = 3</i></p> <p>Replaces -d option(s).</p> <p><i>Optional</i> SubStanza specifying the level of debugging desired (1-3). Used internally to determine the quantity and detail of syslog messages from the PFTP Daemon.</p>
SubStanza	<p>Non-Parallel HostName = <value></p> <p><i>E.g. Non-Parallel HostName = aixrahe.sandia.gov</i></p> <p>Replaces -h option and hpss_option HOSTname in ftpaccess.</p> <p><i>Optional</i> SubStanza specifying the network interface to be used for data transferred between the PFTPD and the Movers when performing non-parallel transfers. Sets the HPSS_HOSTNAME environment variable for the Client API. (HPSS only.)</p>
SubStanza	<p>PFTP Debug Port = <value></p> <p><i>E.g. PFTP Debug Port = 6666</i></p> <p>Replaces -p<port> option.</p> <p><i>Optional</i> SubStanza specifying a port to be used by the HPSS PFTP daemon. Used only when initiating the daemon manually (rather than using inetd/xinetd). May be left on – will not interfere with normal operations</p>
SubStanza	<p>Default Time Out = <value></p> <p><i>E.g. Default Time Out = 1500</i></p> <p>Replaces -t option and hpss_option DTO time in ftpaccess.</p> <p><i>Optional</i> SubStanza specifying the default timeout in seconds.</p>
SubStanza	<p>Default Umask = <value></p> <p><i>E.g. Default Umask = 077</i></p> <p>Replaces -u option and hpss_option UMASK octal in ftpaccess.</p> <p><i>Optional</i> SubStanza specifying the default umask in octal.</p>

SubStanza	<p>Client API Verbose Value = <value></p> <p><i>E.g. Client API Verbose Value = 1</i></p> <p>Replaces -v option(s).</p> <p><i>Optional</i> SubStanza specifying the level of HPSS Client API Logging to use (1-3). The Client API will perform logging specified by the HPSS_DEBUG environment variable in a file specified by the HPSS_DEBUGPATH environment variable</p> <p>(default name is /var/hpss/ftp/adm/hpss_ftpd.log.) The default value is 1. (HPSS only.)</p>
SubStanza	<p>Disallow User Setting of COS</p> <p><i>E.g. Disallow User Setting of COS</i></p> <p>Replaces -C option.</p> <p><i>Optional</i> SubStanza to disable the ability of clients to explicitly set the Class of Service for new files (via the “site setcos” command). Not recommended.</p>
SubStanza	<p>Keytab File Name = <value></p> <p><i>E.g. Keytab File Name = /etc/v5srvtab</i></p> <p>Replaces -K option and hpss_option KTAB string in ftpassess.</p> <p><i>Optional</i> SubStanza specifying the {Path}/{Name} of the Kerberos keytab file containing the service principals for the Kerberized PFTP Servers. Default: /etc/v5srvtab. (HPSS only.)</p>
SubStanza	<p>HPSS Realm Name = <value></p> <p><i>E.g. HPSS Realm Name = FIRE.CLEARLAKE.IBM.COM</i></p> <p><i>Optional</i> SubStanza setting the Realm name for the PFTP Daemon.</p>
SubStanza	<p>Use Port Range</p> <p><i>E.g. Use Port Range</i></p> <p>Replaces -R option.</p> <p><i>Optional</i> SubStanza setting the HPSS PFTP daemon to use a specified port range for connections to the HPSS movers for parallel transfer functions. The actual port range is specified in the mover specific configuration record. Refer to Section 6.8.13: <i>Configure the PVR Specific Information</i> (page 373) for more information. (HPSS only.) NOTE: This is ignored in passive mode!</p>

SubStanza	<p>Maximum Time Out = <value></p> <p>Value: Time in seconds</p> <p><i>E.g. Maximum Time Out = 86400</i></p> <p>Replaces -T option and hpss_option MTO time in ftpaccess.</p> <p><i>Optional</i> SubStanza specifying the maximum timeout in seconds.</p>
SubStanza	<p>Use UDP ONLY</p> <p><i>E.g. UDP ONLY</i></p> <p>Replaces -U option.</p> <p><i>Optional</i> SubStanza specifying use of UDP RPCs Only. Sets the environment variable: RPC_SUPPORTED_PROTSEQS=ncadg_ip_udp (HPSS only.)</p>
SubStanza	<p>Use Extended Registry Attributes</p> <p><i>E.g. Use Extended Registry Attributes</i></p> <p>Replaces -X option.</p> <p><i>Optional</i> SubStanza specifying use of the LDAP Registry for authentication (bypassing the passwd file) and use of the HPSS.homedir and HPSS.gecos Extended Registry Attributes (ERAs) for the users home directory and accounting fields (if they are filled.)</p>
SubStanza	<p>Print Performance Data</p> <p><i>E.g. Print Performance Data</i></p> <p><i>Optional</i> SubStanza specifying the printing of additional performance numbers. (non-HPSS PFTP daemon only.)</p>
SubStanza	<p>Number of Ports = <value></p> <p>Values: 1 - 64</p> <p><i>E.g. Number of Ports = 16</i></p> <p><i>Optional</i> SubStanza specifying the maximum stripe width allowed. (non-HPSS PFTP daemon only.)</p>
SubStanza	<p>IOR Listen Port = <value></p> <p><i>E.g. IOR Listen Port = 19199</i></p> <p><i>Optional</i> SubStanza specifying the port to be used for the PDATA_PUSH protocol. (HPSS only.)</p>

SubStanza	<p>PortRange = <value></p> <p><i>E.g. PortRange = ncadg_ip_udp[10100-12100]:ncacn_ip_tcp[10100-12100]</i></p> <p><i>Optional</i> SubStanza specifying the port range to be used for the non-HPSS parallel FTP daemon which is necessary for parallel transfers. NOTE: This is ignored for passive listings, etc.</p>
SubStanza	<p>Socket Buffer Size = <value></p> <p>Values: Viable Socket Sizes</p> <p><i>E.g. Socket Buffer Size = 1MB</i></p> <p><i>Optional</i> SubStanza specifying the socket buffer size. (non-HPSS PFTP daemon only.)</p>
SubStanza	<p>Set COS Based on Filesize</p> <p>Values: Pathname/filename</p> <p><i>E.g. Set COS Based on Filesize</i></p> <p><i>Optional</i> SubStanza specifying to set the COS from the FileSize Options table. Default: Ignore the COS in the table.</p>
SubStanza (Compound)	<p>FileSize Options = {</p> <p><i>E.g. FileSize Options = {</i></p> <p><i>Optional</i> SubStanza specifier.</p> <p>Must be terminated with a matching “}”</p> <p>See notes below.</p>
Section (Compound)	<p><value> = {</p> <p><i>E.g. 1MB = {</i></p> <p><i>Optional</i> Section specifier.</p> <p>Must be terminated with a matching “}”</p> <p>See notes below.</p>
SubSection	<p>BlockSize = <value></p> <p><i>E.g. BlockSize = 512KB</i></p> <p><i>Optional</i> SubSection specifying the size of data blocks to be used based on file size. (Has no meaning for the HPSS PFTP daemon.)</p>
SubSection	<p>StripeWidth = <value></p> <p><i>E.g. StripeWidth = 0</i></p> <p><i>Optional</i> SubSection specifying the stripe width to be used based on file size. 0 means use the Core Server Value (HPSS PFTP daemon) or use the default (Non-HPSS PFTP daemon).</p>

SubSection	<p>COS = <value></p> <p><i>E.g. COS = 2</i></p> <p><i>Optional</i> SubSection specifying the Class of Service to be used based on file size. 0 means allow the Core Server to determine the optimal COS. (Has no meaning for the Non-HPSS PFTD daemon.)</p>
SubStanza	<p><nodename> Service Name = <canonicalname></p> <p><i>E.g. sunrahe Service Name = sunrahe.sandia.gov</i></p> <p><i>Optional</i> SubStanza specifying the service name to be used by the PFTP daemon node when acquiring credentials. Needed when the servername in the keytab is different from that obtained by gethostname(). Use multiple entries when this file is common to multiple PFTP daemons. Useful particularly for clusters and systems having multiple names. One or the other of host/servicename@realm or ftp/servicename@realm must exist in the Kerberos KDC and in the <code>/etc/v5srvtab</code> for the PFTP Server executing on the machine mymachine.ssm.com (Non-HPSS PFTP daemon only.)</p>
SubStanza	<p>Use System Password Files = <value></p> <p><i>E.g. Use System Password Files = TRUE</i></p> <p>SubStanza specifying that the system password files (<code>/etc/passwd</code>, <code>/etc/group</code>, <code>/etc/shadow</code>) should be used. Should be specified explicitly. TRUE and FALSE are case sensitive!</p>
SubStanza	<p>PFTP Password File = <value></p> <p>Value: Pathname/Filename</p> <p><i>E.g. PFTP Password File = /var/hpss/etc/passwd</i></p> <p><i>Optional</i> SubStanza used to specify the file containing the users password information. <code>/var/hpss/etc/passwd</code> is the default.</p>
SubStanza	<p>PFTP Shadow File = <value></p> <p>Value: Pathname/Filename</p> <p><i>E.g. PFTP Shadow File = /var/hpss/etc/pftp_shadow</i></p> <p><i>Optional</i> SubStanza used to specify the file containing the users protected password information. This should be specified if USERNAME/PASSWORD authentication is in effect.</p>

SubStanza	<p>PFTP Group File = <value></p> <p>Value: Pathname/Filename</p> <p><i>E.g. PFTP Group File = /var/hpss/etc/pftp_groups</i></p> <p><i>Optional</i> Substanza used to specify the file containing the group information for PFTP clients. Default is /var/hpss/etc/group.</p>
SubStanza	<p>Primary Authentication Mechanism = <value></p> <p>Values: KRB5, IBM_HPSS_PKEY, GSI, UNIX</p> <p><i>E.g. Primary Authentication Mechanism = KRB5</i></p> <p><i>Optional</i> Substanza used to specify the default Authentication mechanism.</p> <p>NOTE: IBM_HPSS_PKEY is NOT implemented.</p>
SubStanza	<p>Primary Authenticator Type = <value></p> <p>Values: KEYTAB, KEYFILE, KEY, PASSWD</p> <p><i>E.g. Primary Authenticator Type = KEYTAB</i></p> <p><i>Optional</i> Substanza used to specify the default Authenticator Type.</p>
SubStanza	<p>Primary Authenticator = <value></p> <p>Values: Pathname/Filename</p> <p><i>E.g. Primary Authenticator = /var/hpss/etc/hpss.keytab</i></p> <p><i>Optional</i> Substanza used to specify the file containing the information to authenticate/authorize the hpssftp principal.</p>
SubStanza (Compound)	<p>Client Authentication Mechanisms = {</p> <p>Must be terminated with a matching “}”</p>
Section (Compound)	<p><Type> = {</p> <p>Types: GSS, GSI, IDENT, USER_PASS</p> <p>Must be terminated with a matching “}”</p>

SubSection	<p>Mapfile Specifier = <value></p> <p>Values: Pathname/filename</p> <p><i>E.g. Mapfile Specifier = /var/hpss/etc/MapfileName</i></p> <p><i>Optional</i> Substanza used to specify a file containing username mappings. A different file can exist for each Authentication Type. This file provides the ability to Authenticate as one user and be authorized as another user (entity account). These files MUST be protected for security reasons. These files should be owned by root and readable/writeable ONLY by root.</p>
SubSection	<p>Default Authorization Mechanism = <value></p> <p>Values: LDAP, UNIX, DBAS (Not Implemented)</p> <p><i>E.g. Default Authorization Mechanism = LDAP</i></p> <p><i>Optional</i> Substanza used to specify the Authorization Mechanism desired. The PFTP Daemon does Authorization internally. If the HPSS system is configured to use LDAP and the PFTP Server is configured to use UNIX, the end user will have to be in both Authorization facilities. DBAS has NOT been implemented in PFTP or in HPSS.</p>
SubSection	<p>Use Site Auth Method = <value></p> <p>Values: CRYPTOCARD, KRB5KDC, SECURID</p> <p><i>E.g. Use Site Auth Method = CRYPTOCARD</i></p> <p><i>Optional</i> Substanza used to specify an "Site Specific" Authentication Mechanism to be used instead (Overloaded) of the UserName/Password mechanism. This option requires a specific recompile of the hpss_pftpd with site specific modules linked in. NOTE: if this option is specified, the UserName/Password Mechanism will NOT use a standard Password.</p>
SubStanza	<p>{Hostname} Service Name = {servicename}</p> <p><i>E.g. mymachine Service Name = fire.clearlake.ibm.com</i></p> <p><i>Optional</i> Substanza used to specify alternate service names for the Kerberos service principals. The value after the equal sign is appended to either "host" or "ftp" to form a service by the name like: host/fire.clearlake.ibm.com@realm</p> <p>Very useful for Computing Clusters and multi-homed systems using a Kerberized PFTP Server.</p>

All SubStanzas are optional. If the optional SubStanza **FileSize Options** = { is included, one or more <value> = { Sections must be included with mandatory SubSections **BlockSize** = ..., **StripeWidth** = ..., and **COS** =

Each <value> = { Section defines the beginning of a range of file sizes to which its settings apply. That range begins with **value**, and extends to the next larger **value** included in the **FileSize options** = { SubStanza. In the example below, the settings in the **1MB** = { Section below apply to files with

sizes in the range [1MB, 2MB).

PFTP Daemon Stanza Example:

```
PFTP Daemon = {
    # Allow the Daemon to take Core Dumps
    ; Allow Core Files
    # Directory to put core files in (Default = .)
    ; Core File Directory = /var/hpss/adm/core
    # Specify the SYSLOG facility to use for all syslog messages
    # except Authentication Messages.
    # Values: DAEMON, LOCAL<0-7>
    # Replaces -sstring option.    Default = DAEMON
    SYSLOG Facility = LOCAL0
    # Disable Transfer Agent Support - Default is on
    # HPSS PFTP Server explicitly sets to disabled
    ; No Transfer Agent Support
    # Specify if the KDC registry is to be used as opposed to the
    # passwd file.  "Use Extended Registry Attributes" are specified
    ; Use the KDC Registry
    # Specify the Base Directory for PFTP Files
    ; FTP Base Directory = /var/hpss
    # Specify the name of the ftpaccess file
    # This becomes {BaseDir}/etc/{ftpaccess} based on the FTP Base
Directory
    ; FTP Access File = ftpaccess
    # Do NOT allow / to be Home Directory (HPSS Only)
Disable Slash Home Directory
    # Terminate session if Home Directory does NOT Exist (HPSS Only)
Disable Access if no Home Directory

    # Disable the Sending of Hints to the Core Server (HPSS Only)
    ; Disable Hints
    # Permit use of the Trusted Hosts File (HPSS Only)
    ; Allow Trusted Hosts Authentication
    # What Principal to use for HPSS FTP (HPSS Only)
    ; HPSS FTP Principal = hpssftp
    # Specify the Keytab containing the FTP principal
    ; FTP Principal Keytab File = /var/hpss/etc/hpss.keytab
    # Allow both Active and Passive Connections (HPSS Only)
    ; Allow Passive Connections
    # Delay for xx seconds to allow dbx attach to the Daemon.
    ; Sleep for Debugger = 5
    # For Credentials-based PFTP, Deny username/password authentication
    ; Must Have Credentials

    # For Credentials-based PFTP, Use the Channel Bindings
    ; Use Channel Bindings
    # Allow the CCC Command to the GSS Daemon (non-HPSS Only)
    # This allow for the Control channel to be in cleartext
    ; Allow CCC Command
    # Set the IO Buffer Size for the HPSS PFTP Daemon
    ; PFTP IO Buffer Size = 1MB
    # Specify the Level of Debugging Desired.
    ; Debug Value = 1
    # For non-Parallel Transfers, specify the Interface (by Name)
    # to use between the PFTP Daemon and the Movers (HPSS Only)
    # Replaces -h option and "hpss_option HOST name" in ftpaccess
    ; Non-Parallel HostName = aixrahe.sandia.gov
    # Specify the Port to be used for Manual PFTP Daemon Startup
    ; PFTP Debug Port = 6666
    # Specification in seconds for the Default Timeout
```

```

; Default Time Out = 1500
# Specify (in octal) the Default umask
; Default Umask = 077
# Specification of the Level of HPSS Client API logging to use ( 0 - 3
)
; Client API Verbose Value = 0
# Do NOT allow the user to specify Classes of Service (HPSS Only)
; Disallow User Setting of COS
# Name the Kerberos Keytab file for "Kerberized" PFTP Daemon (HPSS
Only)
; Keytab File Name = /etc/v5srvtab
# Name of the Realm associated with the PFTP Daemon (HPSS Only)
; HPSS RealmName = realm.com
# Specify to make the Client API use the port range specified
; Use Port Range
# Specification in seconds for the Maximum Timeout
; Maximum Time Out = 86400
# Disallow the use of TCP (RPC_SUPPORTED_PROTSEQS=ncadg_ip_udp)
; Use UDP ONLY
# Use the Extended Registry Attributes if they Exist (HPSS.conf)
; Use Extended Registry Attributes
# Print additional Performance Numbers (non-HPSS PFTP Daemon Only)
; Print Performance Data
# Specify the Maximum Stripe Width Allowed (non-HPSS PFTP Daemon Only)
; Number of Ports = 16
# Port to be used for the PDATA_PUSH Protocol. (HPSS Only)
; IOR Listen Port = 19199
# The Port Range to be used for the non-HPSS Parallel FTP Daemon
# which is necessary for Parallel Transfers (non-HPSS PFTP Daemon
Only)
; PortRange = ncadg_ip_udp[10100-12100]:ncacn_ip_tcp[10100-12100]
# The Socket Buffer Size (non-HPSS PFTP Daemon Only)
; Socket Buffer Size = 1MB
# Uncomment next line to use the COS from the FileSize Options
# The default is to ignore the COS specification in the Table.
; Set COS Based on Filesize
# Specify Blocksizes, StripeWidths, and COSs to use based on file size
# COS has no meaning to the Non-HPSS PFTP Daemon
# COS = 0 means allow the BitFile Server to determine the optimal COS
# BlockSize has no meaning to the HPSS PFTP Daemon Only
# StripeWidth = 0 means use the Bitfile Server Value (HPSS PFTP
Daemon)
# or use the default (Non-HPSS PFTP Daemon)
; FileSize Options = {
# Files greater than or equal to this value and less than
# any other value in the table use these Settings
# e.g., 2MB <= filesize < 10MB
; 1MB = {
; BlockSize = 512KB
; StripeWidth = 0
; COS = 2
; }
; 2MB = {
; BlockSize = 2MB
; StripeWidth = 4
; COS = 0
; }
; 10MB = {
; BlockSize = 2MB
; StripeWidth= 0
; COS = 0
; }
; }

```

```

; 100MB = {
    ; BlockSize = 4MB
    ; StripeWidth= 0
    ; COS = 0
; }
; 1GB = {
    ; BlockSize = 8MB
    ; StripeWidth= 0
    ; COS = 0
; }
; }
; }

# Use the System Password file routines (TRUE or FALSE)
# The Default for PFTP is FALSE (Case Sensitive!)
Use System Password Files = FALSE

# Path and Name for the PFTP Password File
PFTP Password File = /var/hpss/etc/passwd

# Path and Name for the PFTP Shadow Password File
# NOTE: PFTP does not currently use the value. It is used ONLY to
# change how the password is looked up! If the site is using
# /etc/passwd and the system running the PFTP Daemon utilizes
# some form of "Shadow" password file to authenticate PFTP users,
# this should be uncommented.
# Do NOT remove the part after the "=" sign.
; PFTP Shadow File = /etc/security/passwd

# Path and Name for the PFTP Group File
; PFTP Group File = /etc/group
# Primary Authentication Type for the FTP Daemon authentication
; Primary Authentication Mechanism = KRB5
# Default: KEYTAB
; Primary Authenticator Type = KEYTAB
# Default: /var/hpss/etc/hpss.keytab
; Primary Authenticator = /var/hpss/etc/hpss.keytab
# Supported Client Authentication Mechanisms
# This mechanism will be used to authenticate the FTP client
# (end-user) with the FTP Daemon
# Values: GSS, GSI, USER_PASS, IDENT
# Default: USER_PASS && "Primary Authentication Mechanism"
# Mapfile Specifier specifies the type and required information
# for Mapping one user name to another. The types include
# "FILE:", "LDAP:", and "DBAS:" The default type is "FILE:"
# For "FILE:" specify the path and name of the file after the ":"
# Default Authorization Mechanism specifies the location of the
# pw_struct info. This may be "UNIX", "LDAP", or "DBAS"
# "DBAS" is NOT currently supported.
# This probably needs to be specified if a name mapping occurs.
# Use Site Auth Method is used to specify that the USER_PASS Mode is
# Overloaded and Hooks have been input to change the behaviour; e.g.,
# USER_PASS will actually use a Crypto Card to Authenticate. This is
# ONLY valid in the USER_PASS clause. NOTE: This is exactly as
# described: if this is specified for USER_PASS, it is impossible
# to allow both username/password AND username/CryptoCard
simultaneously.
Client Authentication Mechanisms = {
; GSS = {
    ; Mapfile Specifier = FILE:/var/hpss/etc/KRB2UnixMapfile
    ; Use Site Auth Method = KRB5KDC
    ; Default Authorization Mechanism = LDAP
; }
; }

```

```

; GSI = {
;   Mapfile Specifier = LDAP:/var/hpss/etc/KRB2UnixMapfile
;   Use Site Auth Method = CryptoCard
;   Default Authorization Mechanism = LDAP
; }
; IDENT = {
;   Mapfile Specifier = FILE:/var/hpss/etc/IDENT2UnixMapfile
;   Use Site Auth Method = SecurId
;   Default Authorization Mechanism = DBAS
; }
USER_PASS = {
;   Use Site Auth Method = KRB5KDC
;   Mapfile Specifier = FILE:/var/hpss/etc/Unix2UnixMapfile
;   Default Authorization Mechanism = UNIX
}
}

# Keytab Hostname Mapping Section
# Syntax:
#   machinename Service Name = canonicalname
# "machinename" is the name returned by a call to gethostname()
# in the PFTP Daemon.
# "canonicalname" is the machine name associated with the Kerberos
# service - usually the fully qualified name as generated by the
# PFTP Client
# Specify "{machinename} Service Name" to set the service name to be
used
# for the PFTP Daemon machine when acquiring creds. This is needed
# when the servername; e.g., host/machinename@realm is different
# between the keytab file and the host/machinename obtained where
# the machinename is obtained by gethostname() (Non-HPSS PFTP Daemon)
# Specify multiple "machinename Service Name" entries if this
# file is common to multiple PFTP Daemon servers.
; aixrahe.sandia.gov Service Name = aixrahe.sandia.gov
; sunrahe Service Name = sunrahe.sandia.gov
}

```

D.6. Transfer Agent Stanza

A large number of options are available for configuring the Transfer Agent and tuning its performance. The Transfer Agent is still under development and the following options are subject to change without prior notification.

Table 17. Transfer Agent Stanza Description

Configuration Type	Abbreviated Description
Stanza (CMPD)	Transfer Agent = { <i>Reserved</i> Stanza specifier. Must be terminated with a matching “}”

SubStanza	<p>Nodeset File = <value></p> <p>Value: Pathname/filename</p> <p><i>E.g. Nodeset File = /var/hpss/etc/PMTA_NodeSets</i></p> <p><i>Optional</i> Substanzaused to specify the location of the Transfer Agent file containing Nodesets.</p>
SubStanza	<p>Node Affinity File = <value></p> <p>Value: Pathname/filename</p> <p><i>E.g. Node Affinity File = /var/hpss/etc/PMTA_NodeAffinity</i></p> <p><i>Optional</i> Substanzaused to specify the location of the Transfer Agent file containing Node Affinities.</p>
SubStanza	<p>Shared FS File = <value></p> <p>Value: Pathname/filename</p> <p><i>E.g. Shared FS File = /var/hpss/etc/PMTA_SharedFilesystems</i></p> <p><i>Optional</i> Substanzaused to specify the location of the Transfer Agent file containing Shared File Systems</p>
SubStanza	<p>Agent File = <value></p> <p>Value: Pathname/filename</p> <p><i>E.g. Agent File = /var/hpss/etc/PMTA_Agents</i></p> <p><i>Optional</i> Substanzaused to specify the location of the Transfer Agent file containing viable Transfer Agent Nodes.</p>
SubStanza	<p>Disabled Node File = <value></p> <p>Value: Pathname/filename</p> <p><i>E.g. Disabled Node File = /var/hpss/log/PMTA_AuditLog</i></p> <p><i>Optional</i> Substanzaused to specify the location of the Transfer Agent Audit log.</p>
SubStanza	<p>Audit Logfile = <value></p> <p>Value: Pathname/filename</p> <p><i>E.g. Audit Logfile = /var/hpss/log/PMTA_AuditLog</i></p> <p><i>Optional</i> Substanzaused to specify the location of the Transfer Agent Audit log.</p>

SubStanza	<p>Debug Logfile = <value></p> <p>Value: Pathname/filename</p> <p><i>E.g. Debug Logfile = /var/hpss/log/PMTA_Debugfile</i></p> <p><i>Optional</i> SubStanza used to specify the location of the Transfer Agent Debugging File.</p>
SubStanza	<p>SYSLOG Facility = <value></p> <p>Values: DAEMON, LOCAL0 ... LOCAL7, NONE</p> <p><i>E.g. SYSLOG Facility = LOCAL0</i></p> <p><i>Optional</i> SubStanza specifying the syslog facility for the HPSS PFTPD. The default syslog facility is LOG_DAEMON (specified by DAEMON - reference: /usr/include/sys/syslog.h). Incorrect specification will default back to LOG_DAEMON.</p>
SubStanza	<p>Allow Uid Mapping = <value></p> <p>Values: off, on</p> <p><i>E.g. Allow Uid Mapping = on</i></p> <p><i>Optional</i> SubStanza specifying whether to allow Transfer Agent Uid Mapping.</p>
SubStanza	<p>Uid Mapfile = <value></p> <p>Value: Pathname/filename</p> <p><i>E.g. Uid Mapfile = /var/hpss/etc/PMTA_UidMapfile</i></p> <p><i>Optional</i> SubStanza used to specify the location of the Transfer Agent Uid Mapfile</p>
SubStanza	<p>Allow Gid Mapping = <value></p> <p>Values: off, on</p> <p><i>E.g. Allow Gid Mapping = on</i></p> <p><i>Optional</i> SubStanza specifying whether to allow Transfer Agent Gid Mapping.</p>
SubStanza	<p>Gid Mapfile = <value></p> <p>Value: Pathname/filename</p> <p><i>E.g. Gid Mapfile = /var/hpss/etc/PMTA_GidMapfile</i></p> <p><i>Optional</i> SubStanza used to specify the location of the Transfer Agent Gid Mapfile</p>

All Stanza and Sub Components are optional.

Transfer Agent Stanza Example:

```

# Parallel Multinode Transfer Agent (PMTA) Section
Transfer Agent = {
    # The (optional) NodeSet File contains named sets of
    # Nodes that can be referred to via the "SET:setname"
    # notation.
    Nodeset File = /var/hpss/etc/ta/nodeset.conf

    # The (optional) Node Affinity file is used to specify
    # groups of nodes are able to communicate in a
    # network whose topology does not support full interconnection
    Node Affinity File = /var/hpss/etc/ta/node_affinity.conf

    # The Shared Filesystem file is used to specify the
    # list of shared filesystem mount points and associated
    # nodes or NodeSets
    Shared FS File = /var/hpss/etc/ta/shared_fs.conf

    # The Agent file contains the list of client hosts and
    # the list of nodes that can be used as Agents from each client
    Agent File = /var/hpss/etc/ta/agent.conf

    # The Disabled Node file is used to disable selection of
    # nodes that are temporarily unavailable. It overrides
    # entries in the Agent file
    Disabled Node File = /var/hpss/etc/ta/disabled_node.conf

    # The Audit Logfile contains an audit trail of all PMTA activity
    Audit Logfile = /tmp/pmta_Audit.log

    # The Debug Logfile contains debugging output. It can
    # be overridden by environment variable settings.
    Debug Logfile = /tmp/pmta_Debug.log

    # The SYSLOG Facility parameter controls logging to syslog. Legal
    # values are "off" or "LOCALn" where n=0-7.
    SYSLOG Facility = off

    # The optional "Allow Uid Mapping" setting defines whether the same
user
    # can have different UIDs on different machines within the site.
    # The default value is NO
    Allow Uid Mapping = NO

    # The optional "Uid Mapfile" setting is the name of the
    # mapping file when "Allow Uid Mapping" is set to "YES"
    Uid Mapfile = /var/hpss/etc/ta/uid_mapfile

    # The optional "Allow Gid Mapping" setting defines whether the same
group
    # can have different GIDs on different machines within the site.
    # The default value is NO
    Allow Gid Mapping = NO

    # The optional "Gid Mapfile" setting is the name of the mapping
    # file used when "Allow Gid Mapping" is set to "YES"
    Gid Mapfile = /var/hpss/etc/ta/gid_mapfile
}
# end of Transfer Agent section

```

D.7. Stanzas Reserved for Future Use

The following stanza names (specifiers) are reserved for future implementation in HPSS and should not be used by application developers.

- Transfer Agent (Partially Implemented)
- Pipe File
- Local File Path
- PSI

Appendix E. hpss_env_defs.h

The HPSS environment variables are defined in /opt/hpss/include/hpss_env_defs.h. These environment variables can be overridden in /var/hpss/etc/env.conf or in the local environment.

```
static env_t  hpss_env_defs[] = {
/*
*****
*
*   HPSS_ROOT           - Root pathname for HPSS Unix top level
*   HPSS_HOST           - Machine host name
*   HPSS_NODE_TYPE     - Node type of current machine
*   HPSS_PATH_INSTALL  - Pathname for HPSS installation
*   HPSS_PATH_BIN      - Pathname for HPSS executables
*   HPSS_PATH_MSG      - Pathname for HPSS message catalog
*   HPSS_PATH_SLASH_BIN - Pathname for /bin
*   HPSS_PATH_SLASH_ETC - Pathname for /etc
*   HPSS_PATH_USR_BIN  - Pathname for /usr/bin
*   HPSS_PATH_USR_SBIN - Pathname for /usr/sbin
*   HPSS_PATH_VAR      - Pathname for HPSS var directory
*   HPSS_USER          - HPSS user name
*   HPSS_USERROOT      - Root user id
*   HPSS_PATH_DB_INSTALL - Pathname for DB bin
*****
*/
    { "HPSS_ROOT",           "/opt/hpss",           NULL},
    { "HPSS_HOST",          "%H",                  NULL},
    { "HPSS_NODE_TYPE",    NULL,                  NULL},
    { "HPSS_PATH_INSTALL",  "/opt/hpss",          NULL},
    { "HPSS_PATH_BIN",     "${HPSS_PATH_INSTALL}/bin",
      NULL},
    { "HPSS_PATH_MSG",     "${HPSS_PATH_INSTALL}/msg/En_US",
      NULL},
    { "HPSS_PATH_SLASH_BIN", "/bin",                NULL},
    { "HPSS_PATH_SLASH_ETC", "/etc",                 NULL},
    { "HPSS_PATH_USR_BIN",  "/usr/bin",            NULL},
    { "HPSS_PATH_USR_SBIN", "/usr/sbin",           NULL},
    { "HPSS_PATH_VAR",     "/var/hpss",           NULL},
    { "HPSS_USER",         "hpss",                 NULL},
    { "HPSS_USERROOT",     "root",                 NULL},
    { "HPSS_PATH_DB_INSTALL", NULL,                  NULL},
    { "HPSS_SYSTEM",       "%S",                  NULL},
    { "HPSS_SYSTEM_VERSION", "%V",                  NULL},
    { "HPSS_HOST_FULL_NAME", "%L",                  NULL},
/*
*****
*
*   HPSS Group names
*   HPSS_GRP_NAME         - HPSS group name
*   HPSS_GRP_NAME_SERVER - HPSS Server group name
*   HPSS_GRP_NAME_CLIENT - HPSS Client group name
*****
*/
    { "HPSS_GRP_NAME",           "hpss",
      NULL},
    { "HPSS_GRP_NAME_SERVER",    "hpsssrvr",
      NULL},

```

```

        { "HPSS_GRP_NAME_CLIENT",          "hpss_client",
NULL},
/*
*****
*
* HPSS Server Principal names
*
*   HPSS_PRINCIPAL           - Principal name for SEC Server
*   HPSS_PRINCIPAL_CORE     - Principal name for CORE Server
*   HPSS_PRINCIPAL_DMG      - Principal name for DMAP Gateway
*   HPSS_PRINCIPAL_FTPD     - Principal name for FTP Daemon
*   HPSS_PRINCIPAL_GK       - Principal name for Gatekeeper Server
*   HPSS_PRINCIPAL_HDM      - Principal name for HDM
*   HPSS_PRINCIPAL_HPSSD    - Principal name for Startup Daemon
*   HPSS_PRINCIPAL_LOG      - Principal name for Log Client and
Daemon
*   HPSS_PRINCIPAL_LS       - Principal name for Location Server
*   HPSS_PRINCIPAL_MOUNTD   - Principal name for Mount Daemon
*   HPSS_PRINCIPAL_MPS      - Principal name for Migration/Purge
Server
*   HPSS_PRINCIPAL_MVR      - Principal name for Mover
*   HPSS_PRINCIPAL_NFSD     - Principal name for NFS Daemon
*   HPSS_PRINCIPAL_PVL      - Principal name for PVL
*   HPSS_PRINCIPAL_PVR      - Principal name for PVR
*   HPSS_PRINCIPAL_SSM      - Principal name for SSM
*   HPSS_PRINCIPAL_ADM_USER - Principal name for primary HPSS
                        administrator principal
*****
*
*/
        { "HPSS_PRINCIPAL",          NULL,          NULL},
        { "HPSS_PRINCIPAL_CORE",     "hpsscore",   NULL},
        { "HPSS_PRINCIPAL_DMG",      "hpssdmg",    NULL},
        { "HPSS_PRINCIPAL_FTPD",     "hpssftp",    NULL},
        { "HPSS_PRINCIPAL_GK",       "hpssgk",     NULL},
        { "HPSS_PRINCIPAL_HDM",      "hpsshdm",    NULL},
        { "HPSS_PRINCIPAL_HPSSD",    "hpsssd",     NULL},
        { "HPSS_PRINCIPAL_LOG",      "hpsslog",    NULL},
        { "HPSS_PRINCIPAL_LS",       "hpssls",     NULL},
        { "HPSS_PRINCIPAL_MOUNTD",   "hpssmntd",   NULL},
        { "HPSS_PRINCIPAL_MPS",      "hpssmps",    NULL},
        { "HPSS_PRINCIPAL_MVR",      "hpssmvr",    NULL},
        { "HPSS_PRINCIPAL_NFSD",     "hpssnfs",    NULL},
        { "HPSS_PRINCIPAL_PVL",      "hpsspvl",    NULL},
        { "HPSS_PRINCIPAL_PVR",      "hpsspvr",    NULL},
        { "HPSS_PRINCIPAL_SSM",      "hpssssm",    NULL},
        { "HPSS_PRINCIPAL_ADM_USER", "${HPSS_USER}", NULL},
/*
*****
*
* HPSS Server Principal UID's
*
*   HPSS_PRINCIPAL_CORE_UID   - Principal UID for CORE Server
*   HPSS_PRINCIPAL_DMG_UID    - Principal UID for DMAP Gateway
*   HPSS_PRINCIPAL_FTPD_UID   - Principal UID for FTP Daemon
*   HPSS_PRINCIPAL_GK_UID     - Principal UID for Gatekeeper Server
*   HPSS_PRINCIPAL_HDM_UID    - Principal UID for HDM
*   HPSS_PRINCIPAL_HPSSD_UID  - Principal UID for Startup Daemon
*   HPSS_PRINCIPAL_LOG_UID    - Principal UID for Log Client and
Daemon
*   HPSS_PRINCIPAL_LS_UID     - Principal UID for Location Server
*   HPSS_PRINCIPAL_MOUNTD_UID  - Principal UID for Mount Daemon

```

```

*      HPSS_PRINCIPAL_MPS_UID      - Principal UID for Migration/Purge
Server
*      HPSS_PRINCIPAL_MVR_UID      - Principal UID for Mover
*      HPSS_PRINCIPAL_NFSD_UID     - Principal UID for NFS Daemon
*      HPSS_PRINCIPAL_NS_UID       - Principal UID for Name Server
*      HPSS_PRINCIPAL_PFS_D_UID    - Principal UID for PFS Daemon
*      HPSS_PRINCIPAL_PVL_UID      - Principal UID for PVL
*      HPSS_PRINCIPAL_PVR_UID      - Principal UID for PVR
*      HPSS_PRINCIPAL_SS_UID       - Principal UID for Storage Server
*      HPSS_PRINCIPAL_SSM_UID      - Principal UID for SSM
*
* NOTE: Principal UID must be in the format of "-uid <number of uid>"
*       For example:
*           { "HPSS_PRINCIPAL_BFS_UID"          "-uid 1234",          NULL},
*
*****
*/
{ "HPSS_PRINCIPAL_CORE_UID",      "",          NULL},
{ "HPSS_PRINCIPAL_DMG_UID",      "",          NULL},
{ "HPSS_PRINCIPAL_FTPD_UID",     "",          NULL},
{ "HPSS_PRINCIPAL_GK_UID",       "",          NULL},
{ "HPSS_PRINCIPAL_HDM_UID",      "",          NULL},
{ "HPSS_PRINCIPAL_HPSSD_UID",    "",          NULL},
{ "HPSS_PRINCIPAL_LOG_UID",      "",          NULL},
{ "HPSS_PRINCIPAL_LS_UID",       "",          NULL},
{ "HPSS_PRINCIPAL_MOUNTD_UID",   "",          NULL},
{ "HPSS_PRINCIPAL_MPS_UID",      "",          NULL},
{ "HPSS_PRINCIPAL_MVR_UID",      "",          NULL},
{ "HPSS_PRINCIPAL_NFSD_UID",     "",          NULL},
{ "HPSS_PRINCIPAL_PVL_UID",      "",          NULL},
{ "HPSS_PRINCIPAL_PVR_UID",      "",          NULL},
{ "HPSS_PRINCIPAL_SSM_UID",      "",          NULL},
/*
*****
*
* HPSS Server Executable Names
*
*      HPSS_EXEC_ACCT              - executable name for Accounting
*      HPSS_EXEC_CORE              - executable name for CORE Server
*      HPSS_EXEC_DMG               - executable name for DMAP Gateway
*      HPSS_EXEC_FTPD              - executable name for FTPD
*      HPSS_EXEC_GK                - executable name for Gatekeeper Server
*      HPSS_EXEC_HPSSD             - executable name for Start Daemon
*      HPSS_EXEC_LOGC              - executable name for Log Client
*      HPSS_EXEC_LOGD              - executable name for Log Daemon
*      HPSS_EXEC_LS                - executable name for Location Server
*      HPSS_EXEC_MOUNTD            - executable name for Mount Daemon
*      HPSS_EXEC_MPS               - executable name for Migration/Purge
Server
*      HPSS_EXEC_MVR               - executable name for Mover
*      HPSS_EXEC_MVR_TCP            - executable name for Mover TCP
*      HPSS_EXEC_NFSD              - executable name for NFS Daemon
*      HPSS_EXEC_PFS_D             - executable name for PFS Daemon
*      HPSS_EXEC_PVL               - executable name for PVL
*      HPSS_EXEC_PVR_AMPEX         - executable name for PVR Ampex
*      HPSS_EXEC_PVR_OPER          - executable name for PVR Operator
*      HPSS_EXEC_PVR_STK           - executable name for PVR STK
*      HPSS_EXEC_PVR_3494          - executable name for PVR 3494
*      HPSS_EXEC_PVR_3495          - executable name for PVR 3495
*      HPSS_EXEC_PVR_LTO           - executable name for PVR LTO
*      HPSS_EXEC_PVR_AML           - executable name for PVR AML

```

```

*      HPSS_EXEC_PVR_SCSI      - executable name for PVR SCSI
*      HPSS_EXEC_SSMSM       - executable name for SSM Storage Manager
*
*****
*/
{ "HPSS_EXEC_ACCT",          "${HPSS_PATH_BIN}/hpss_acct",
  NULL},
{ "HPSS_EXEC_CORE",        "${HPSS_PATH_BIN}/hpss_core",
  NULL},
{ "HPSS_EXEC_DMG",         "${HPSS_PATH_BIN}/hpss_dmg",
  NULL},
{ "HPSS_EXEC_FTPD",        "${HPSS_PATH_BIN}/hpss_pftpd",
  NULL},
{ "HPSS_EXEC_GK",          "${HPSS_PATH_BIN}/hpss_gk",
  NULL},
{ "HPSS_EXEC_HPSSD",       "${HPSS_PATH_BIN}/hpssd",
  NULL},
{ "HPSS_EXEC_LOGC",        "${HPSS_PATH_BIN}/hpss_logc",
  NULL},
{ "HPSS_EXEC_LOGD",        "${HPSS_PATH_BIN}/hpss_logd",
  NULL},
{ "HPSS_EXEC_LS",         "${HPSS_PATH_BIN}/hpss_ls",
  NULL},
{ "HPSS_EXEC_MOUNTD",     "${HPSS_PATH_BIN}/hpss_mnt",
  NULL},
{ "HPSS_EXEC_MPS",        "${HPSS_PATH_BIN}/hpss_mps",
  NULL},
{ "HPSS_EXEC_MVR",         "${HPSS_PATH_BIN}/hpss_mvr",
  NULL},
{ "HPSS_EXEC_MVR_TCP",    "${HPSS_PATH_BIN}/hpss_mvr_tcp",
  NULL},
{ "HPSS_EXEC_NFSD",       "${HPSS_PATH_BIN}/hpss_nfs",
  NULL},
{ "HPSS_EXEC_PVL",        "${HPSS_PATH_BIN}/hpss_pvl",
  NULL},
{ "HPSS_EXEC_PVR_AMPEX",  "${HPSS_PATH_BIN}/hpss_pvr_ampex",
  NULL},
{ "HPSS_EXEC_PVR_OPER",   "$
{HPSS_PATH_BIN}/hpss_pvr_operator",
  NULL},
{ "HPSS_EXEC_PVR_STK",    "${HPSS_PATH_BIN}/hpss_pvr_stk",
  NULL},
{ "HPSS_EXEC_PVR_3494",   "${HPSS_PATH_BIN}/hpss_pvr_3494",
  NULL},
{ "HPSS_EXEC_PVR_3495",   "${HPSS_PATH_BIN}/hpss_pvr_3495",
  NULL},
{ "HPSS_EXEC_PVR_LTO",    "${HPSS_PATH_BIN}/hpss_pvr_lto",
  NULL},
{ "HPSS_EXEC_PVR_AML",    "${HPSS_PATH_BIN}/hpss_pvr_aml",
  NULL},
{ "HPSS_EXEC_PVR_SCSI",   "${HPSS_PATH_BIN}/hpss_pvr_scsi",
  NULL},
{ "HPSS_EXEC_SSMSM",     "${HPSS_PATH_BIN}/hpss_ssmsm",
  NULL},
/*
*****
*
* Utilities need by SSM
*
*      HPSS_EXEC_ACL_EDIT   - Pathname for the acl_edit utility
*      HPSS_EXEC_CDSCP     - Pathname for the cdscp utility

```



```

*      HPSS_EXEC_DELOG          - Pathname for the delog utility
*      HPSS_EXEC_RECLAIM       - Pathname for the reclaim utility
*      HPSS_EXEC_REPACK       - Pathname for the repack utility
*
*****
*/
    { "HPSS_EXEC_ACL_EDIT",      "${HPSS_PATH_SLASH_BIN}/acl_edit",
      NULL},
    { "HPSS_EXEC_CDSCP",        "${HPSS_PATH_SLASH_BIN}/cdscp",
      NULL},
    { "HPSS_EXEC_DELOG",        "${HPSS_PATH_BIN}/hpss_delog",
      NULL},
    { "HPSS_EXEC_RECLAIM",      "${HPSS_PATH_BIN}/reclaim",
      NULL},
    { "HPSS_EXEC_REPACK",      "${HPSS_PATH_BIN}/repack",
      NULL},
/*
*****
* Logging Unix files
*
*      HPSS_PATH_LOG           - unix path name for logging files
*      HPSS_UNIX_LOCAL_LOG     - local log file
*****
*/
    { "HPSS_PATH_LOG",          "${HPSS_PATH_VAR}/log", NULL},
    { "HPSS_UNIX_LOCAL_LOG",    "${HPSS_PATH_LOG}/local.log",
      NULL},
/*
*****
* Accounting Unix files
*
files      HPSS_PATH_ACCT       - unix path name for accounting
*
*      HPSS_UNIX_ACCT_CHECKPOINT - checkpoint file
*      HPSS_UNIX_ACCT_REPORT     - report file
*      HPSS_UNIX_ACCT_COMMENTARY - commentary file
*****
*/
    { "HPSS_PATH_ACCT",         "${HPSS_PATH_VAR}/acct",
      NULL},
    { "HPSS_UNIX_ACCT_CHECKPOINT", "$
{HPSS_PATH_ACCT}/acct_checkpoint",
      NULL},
    { "HPSS_UNIX_ACCT_REPORT",   "${HPSS_PATH_ACCT}/acct_report",
      NULL},
    { "HPSS_UNIX_ACCT_COMMENTARY", "$
{HPSS_PATH_ACCT}/acct_commentary",
      NULL},
/*
*****
* NFS Unix files
*
*      HPSS_PATH_NFS           - unix path name for NFS files
*      HPSS_UNIX_NFS_EXPORTS   - exports file
*      HPSS_UNIX_NFS_CREDMAP    - credmap file
*      HPSS_UNIX_NFS_CACHEFILE - cache file

```

```

*      HPSS_UNIX_NFS_CHECKPOINT          - checkpoint
*      HPSS_NFS_DISABLE_JUNCTIONS       - disable traversal of junctions
*      HPSS_MNT_DISABLE_JUNCTIONS       - disable mounting junctions
*****
*/
    { "HPSS_PATH_NFS",                    "${HPSS_PATH_VAR}/nfs", NULL},
    { "HPSS_UNIX_NFS_EXPORTS",           "${HPSS_PATH_NFS}/exports",
      NULL},
    { "HPSS_UNIX_NFS_CREDMAP",           "${HPSS_PATH_NFS}/credmap.nfs",
      NULL},
    { "HPSS_UNIX_NFS_CACHEFILE",         "${HPSS_PATH_NFS}/cachefile.nfs",
      NULL},
    { "HPSS_UNIX_NFS_CHECKPOINT",        "${HPSS_PATH_NFS}/checkpoint.nfs",
      NULL},
    { "HPSS_NFS_DISABLE_JUNCTIONS",      "TRUE",                      NULL},
    { "HPSS_MNT_DISABLE_JUNCTIONS",      "TRUE",                      NULL},
/*
*****
*
* MPS Unix files
*
*      HPSS_PATH_MPS                     - unix path name for MPS files
*      HPSS_UNIX_MPS_REPORT              - report file
*****
*/
    { "HPSS_PATH_MPS",                   "${HPSS_PATH_VAR}/mps", NULL},
    { "HPSS_UNIX_MPS_REPORT",            "",                          NULL},
/*
*****
*
* Gatekeeper Unix files
*
*      HPSS_PATH_GK                      - unix path name for Gatekeeping
files
*      HPSS_UNIX_GK_SITE_POLICY          - site policy file
*****
*/
    { "HPSS_PATH_GK",                    "${HPSS_PATH_VAR}/gk", NULL},
    { "HPSS_UNIX_GK_SITE_POLICY",        "${HPSS_PATH_GK}/gksitepolicy",
      NULL},
/*
*****
*
* Database Info
*
*      HPSS_GLOBAL_DB_NAME               - Default name of the Global Data Base
*      HPSS_SUBSYS_DB_NAME               - Default name for the Subsystem Data Base
*      HPSS_MM_SCHEMA_NAME               - Default schema name
*      HPSS_MM_LOW_WATER_MARK            - default low water mark for DB connections
*      HPSS_MM_HIGH_WATER_MARK           - default high water mark for DB
connections
*      HPSS_MM_CACHE                     - if "off", mm lib won't do stmt caching
*      HPSS_SERVER_DB_GROUP              - DB auth group identity used by HPSS
servers
*      HPSS_SERVER_DB_KEYTAB             - DB connection keytab used by HPSS servers
*****
*/
    { "HPSS_DB_INSTANCE_OWNER",          "hpssdb",                    NULL},
    { "HPSS_GLOBAL_DB_NAME",             "cfg",                        NULL},

```

```

    { "HPSS_SUBSYS_DB_NAME",          "subsys",          NULL},
    { "HPSS_MM_SCHEMA_NAME",         "HPSS",           NULL},
    { "HPSS_MM_LOW_WATER_MARK",      "1",              NULL},
    { "HPSS_MM_HIGH_WATER_MARK",     "5",              NULL},
    { "HPSS_MM_CACHE",               "",                NULL},
    { "HPSS_SERVER_DB_GROUP",        "HPSSSRVR",      NULL},
    { "HPSS_SERVER_DB_KEYTAB",       "${HPSS_PATH_ETC}/mm.keytab",
      NULL},
/*
*****
*
* Descriptive Names
*
* We use HPSS_HOST_TMP instead of HPSS_HOST because HPSS_HOST is
* being set in hpss_env using 'hostname', and so it always gets the
* long name. The default "%H" definition of HPSS_HOST inside
* hpss_env_defs.h gets overridden. But the long hostname is often
* too long for the very short descriptive name field. Force it
* to short name here. Unless, of course, user overrides our new
* HPSS_HOST_TMP in hpss_env...
*
* So to help a little more, we also provide short forms of the
* three descriptive names (logc, mvr, startup daemon) which
incorporate
* the hostname.
*
* HPSS_DESC_CORE           - Descriptive name - Core Server
* HPSS_DESC_DMG           - Descriptive name - DMAP Gateway
* HPSS_DESC_FTPD          - Descriptive name - FTP Daemon
* HPSS_DESC_GK            - Descriptive name - Gatekeeper
Server
* HPSS_DESC_HPSSD         - Descriptive name - Startup Daemon
* HPSS_DESC_HPSSD_SHORT   - Descriptive name - Start Dem,
short
* HPSS_DESC_LOGC          - Descriptive name - Log Client
* HPSS_DESC_LOGC_SHORT   - Descriptive name - Log Client,
short
* HPSS_DESC_LOGD          - Descriptive name - Log Daemon
* HPSS_DESC_LS            - Descriptive name - Location
Server
* HPSS_DESC_MM            - Descriptive name - Metadata
Monitor
* HPSS_DESC_MOUNTD       - Descriptive name - Mount Daemon
* HPSS_DESC_MPS           - Descriptive name - MPS
* HPSS_DESC_MVR           - Descriptive name - Mover
* HPSS_DESC_MVR_SHORT     - Descriptive name - Mover, short
* HPSS_DESC_NFSD          - Descriptive name - NFS Daemon
* HPSS_DESC_PFS           - Descriptive name - PFS
* HPSS_DESC_PVL           - Descriptive name - PVL
* HPSS_DESC_PVR_AMPEX     - Descriptive name - PVR - Ampex
* HPSS_DESC_PVR_OPER      - Descriptive name - PVR - Operator
* HPSS_DESC_PVR_STK       - Descriptive name - PVR - STK
* HPSS_DESC_PVR_STK_RAIT  - Descriptive name - PVR - STK RAIT
* HPSS_DESC_PVR_3494      - Descriptive name - PVR - 3494
* HPSS_DESC_PVR_3495      - Descriptive name - PVR - 3495
* HPSS_DESC_PVR_LTO       - Descriptive name - PVR - LTO
* HPSS_DESC_PVR_AML       - Descriptive name - PVR - AML
* HPSS_DESC_PVR_SCSI      - Descriptive name - PVR - SCSI
* HPSS_DESC_SSM           - Descriptive name - SSM System
Manager
*****
*

```

```

*/
    { "HPSS_HOST_TMP",                "%H",                NULL},
    { "HPSS_DESC_CORE",              "Core Server",      NULL},
    { "HPSS_DESC_DMG",               "DMAP Gateway",    NULL},
    { "HPSS_DESC_FTPD",              "FTP Daemon",      NULL},
    { "HPSS_DESC_GK",                "Gatekeeper",      NULL},
    { "HPSS_DESC_HPSSD",              "Startup Daemon ($
{HPSS_HOST_TMP})",  NULL},
    { "HPSS_DESC_HPSSD_SHORT",        "Startup Daemon",  NULL},
    { "HPSS_DESC_LOGC",               "Log Client ($ {HPSS_HOST_TMP})",
NULL},
    { "HPSS_DESC_LOGC_SHORT",         "Log Client",      NULL},
    { "HPSS_DESC_LOGD",               "Log Daemon",      NULL},
    { "HPSS_DESC_LS",                 "Location Server",  NULL},
    { "HPSS_DESC_MM",                 "Metadata Monitor",  NULL},
    { "HPSS_DESC_MOUNTD",             "Mount Daemon",    NULL},
    { "HPSS_DESC_MPS",               "Migration/Purge Server",
NULL},
    { "HPSS_DESC_MVR",               "Mover ($ {HPSS_HOST_TMP})",
NULL},
    { "HPSS_DESC_MVR_SHORT",          "Mover",           NULL},
    { "HPSS_DESC_NFSD",              "NFS Daemon",      NULL},
    { "HPSS_DESC_PVL",               "PVL",             NULL},
    { "HPSS_DESC_PVR_AMPEX",          "Ampex PVR",       NULL},
    { "HPSS_DESC_PVR_OPER",          "Operator PVR",    NULL},
    { "HPSS_DESC_PVR_STK",           "STK PVR",         NULL},
    { "HPSS_DESC_PVR_STK_RAIT",       "STK RAIT PVR",    NULL},
    { "HPSS_DESC_PVR_3494",          "3494 PVR",        NULL},
    { "HPSS_DESC_PVR_3495",          "3495 PVR",        NULL},
    { "HPSS_DESC_PVR_LTO",           "3584 LTO PVR",    NULL},
    { "HPSS_DESC_PVR_AML",           "AML PVR",         NULL},
    { "HPSS_DESC_PVR_SCSI",          "SCSI PVR",        NULL},
    { "HPSS_DESC_SSM$M",             "SSM System Manager",  NULL},

/*
*****
*
* System Manager Specific
*
*   HPSS_PATH_SSM                - unix path name for data server files
*   HPSS_SSM_ALARMS              - File to store SSM Alarms/Events
*                                   NULL -> SM will store internally
*   HPSS_SSM_ALARMS_DISPLAY      - Number of SSM Alarms/Events to
display/store
*   HPSS_SSM_ALARMS_GET          - Number of SSM Alarms/Events to get at one
time
*   HPSS_SSM_COUNTRY             - Country for Java internationalization
*   HPSS_SSM_LANGUAGE            - Language for Java internationalization
*   HPSS_SSM_SERVER_LISTEN_PORT
*                                   - Port the SM is listening on for client RPCs
*                                   If 0, port will be chosen by portmapper
*   HPSS_HELP_FILES_PATH         - HPSS Help files install path
*   HPSS_HELP_URL_TYPE           - HPSS Help files URL type
*   HPSSGUI_SM_HOST_NAME         - host SM is on
*   HPSSADM_SM_HOST_NAME         - host SM is on
*   HPSSGUI_SM_PORT_NUM          - port SM is on
*   HPSSADM_SM_PORT_NUM          - port SM is on
*   HPSSGUI_RPC_PROT_LEVEL       - rpc protection level used for SM
communication
*   HPSSADM_RPC_PROT_LEVEL       - rpc protection level used for SM
communication
*   HPSSGUI_UI_WAIT_TIME         - Time the GUI will wait at the SM for

```

```

updates
*      HPSSSSM_UI_MO_RATE      - Time the GUI will wait between MO update
tries
*      HPSSSSM_UI_LIST_RATE   - Time the GUI will wait between List
update tries
*      HPSSSSM_UI_ALARM_RATE  - Time the GUI will wait between Alarm
update tries
*      HPSSGUI_SEC_MECH       - security mechanism used for SM
communication
*      HPSSADM_SEC_MECH       - security mechanism used for SM
communication
*      HPSSGUI_USER_CFG_PATH  - Directory which holds GUI config files
*      HPSSADM_USER_CFG_PATH  - Directory which holds ADM config files
*
*      HPSS_SSMUSER_JAVA_POLICY- Java policy file for SSM GUI and hpssadm
*
*      JAVA_ROOT               - top level where Java is installed
*      JAVA_BIN                 - location of tools like the Java
interpreter
*
*      JAVA_CLS1                - location of hpss ssm Java classes
*      HPSS_SSM_CLASSPATH       - runtime search path for Java classes
*
*      The JAVA_HOME variable is used by several HPSS scripts to set the
*      command execution PATH and other variables.
*
*      The SM attempts to throttle the connection attempts to other servers.
It
*      will attempt to reconnect to each server every
*      HPSS_SM_SRV_CONNECT_INTERVAL_MIN seconds until the number of failures
for
*      that server has reached HPSS_SM_SRV_CONNECT_FAIL_COUNT. After the
failure
*      count has been reached the SM will only try to reconnect to the server
*      every HPSS_SM_SRV_CONNECT_INTERVAL_MAX seconds until a successful i
*      connection is made at which time the connection interval for the
server
*      will be set back to HPSS_SM_SRV_CONNECT_INTERVAL_MIN.
*
*      HPSS_SM_SRV_CONNECT_FAIL_COUNT - Number of connection failures to
a
*      server before the HPSS_SM_SRV_CONNECT_INTERVAL_MAX
*      interval takes affect
*      HPSS_SM_SRV_CONNECT_INTERVAL_MIN - Interval between attempting
server
*      connections when HPSS_SM_SERVER_CONNECT_FAIL_COUNT
*      has not yet been reached (seconds)
*      HPSS_SM_SRV_CONNECT_INTERVAL_MAX - Interval between server
connections
*      when HPSS_SM_SERVER_CONNECT_FAIL_COUNT has been
*      reached without a successful connection (seconds)
*      HPSS_SM_SRV_LIST_UPDATE_INTERVAL - Frequency at which the SM
updates
*      the server list (seconds)
*      HPSS_SM_SRV_MONITOR_THREADS   - Number of threads created to
monitor
*      server connections
*      HPSS_SM_CLNT_TPOOL_SIZE       - Thread Pool Size used by the
System
*      Manager client interface
*      HPSS_SM_SRV_MAX_CONNECTIONS   - Number of HPSS server
connections

```

```

*
*           to maintain at once. If this number of connections
is
*
*           exceeded, then old connections will be close to
*
*           maintain this number of connections
*
*****
*/
{ "HPSS_PATH_SSM",           "${HPSS_PATH_VAR}/ssm", NULL},
{ "HPSS_SSM_ALARMS",        NULL, NULL},
{ "HPSS_SSM_ALARMS_DISPLAY", "2000", NULL},
{ "HPSS_SSM_ALARMS_GET",    "500", NULL},
{ "HPSS_SSM_COUNTRY",       "US", NULL},
{ "HPSS_SSM_LANGUAGE",      "en", NULL},
{ "HPSS_SSM_SERVER_LISTEN_PORT", "0", NULL},
{ "HPSS_HELP_FILES_PATH",    "${HPSS_PATH_VAR}/doc", NULL},
{ "HPSS_HELP_URL_TYPE",     "file:", NULL},
{ "HPSSGUI_SM_HOST_NAME",   "${HPSS_HOST}", NULL},
{ "HPSSADM_SM_HOST_NAME",   "${HPSS_HOST}", NULL},
{ "HPSSGUI_SM_PORT_NUM",    "536870913:1", NULL},
{ "HPSSADM_SM_PORT_NUM",    "536870913:1", NULL},
{ "HPSSGUI_RPC_PROT_LEVEL", "${HPSS_RPC_PROT_LEVEL}",
  NULL},
{ "HPSSADM_RPC_PROT_LEVEL", "${HPSS_RPC_PROT_LEVEL}",
  NULL},
{ "HPSSSSM_UI_WAIT_TIME",   "${SSM_UI_WAIT_TIME}", NULL},
{ "HPSSSSM_UI_MO_RATE",     "${SSM_UI_MO_RATE}", NULL},
{ "HPSSSSM_UI_LIST_RATE",   "${SSM_UI_LIST_RATE}", NULL},
{ "HPSSSSM_UI_ALARM_RATE",  "${SSM_UI_ALARM_RATE}", NULL},
{ "HPSSGUI_SEC_MECH",       NULL, NULL},
{ "HPSSADM_SEC_MECH",       NULL, NULL},
{ "HPSSGUI_USER_CFG_PATH",  "${HPSS_PATH_SSM}/hpss-ssm-prefs",
  NULL},
{ "HPSSADM_USER_CFG_PATH",  "${HPSS_PATH_SSM}/hpss-ssm-prefs",
  NULL},
{ "HPSS_SSMUSER JAVA POLICY",
  "${HPSS_PATH_VAR}/ssm/java.policy.ssmuser", NULL},
{ "JAVA_ROOT",
  "/usr/java/j2sdk1.4.1_01", NULL},
{ "JAVA_BIN",               "${JAVA_ROOT}/bin", NULL},
{ "JAVA_CLS1",
  "${HPSS_ROOT}/bin/hpss.jar", NULL},
{ "HPSS_SSM_CLASSPATH",
  "${JAVA_CLS1}", NULL},
{ "HPSS_SM_SRV_CONNECT_FAIL_COUNT", "3", NULL},
{ "HPSS_SM_SRV_CONNECT_INTERVAL_MIN", "20", NULL},
{ "HPSS_SM_SRV_CONNECT_INTERVAL_MAX", "60", NULL},
{ "HPSS_SM_SRV_LIST_UPDATE_INTERVAL", "5", NULL},
{ "HPSS_SM_SRV_MONITOR_THREADS", "10", NULL},
{ "HPSS_SM_CLNT_TPOOL_SIZE", "20", NULL},
{ "HPSS_SM_SRV_MAX_CONNECTIONS", "50", NULL},
/*
*****
*
* CLAPI Specific
*
*   HPSS_API_HOSTNAME           - Used to control the network interface
*                               selected for data transfers.
*
*   HPSS_API_DESC_NAME          - Used in HPSS logging messages if the
*                               logging feature is enabled
*
*   HPSS_API_DEBUG_PATH         - Used to direct debug output messages

```

```

*      HPSS_API_MAX_CONN          - Defines the number of connections
that
*
*      are supported by the Client API
within
*
*      a single client process
*      HPSS_API_DEBUG            - Used to enable debug messages
*      HPSS_API_RETRIES          - Used to control the number of retries
*
*      when operations fail with a
"retryable"
*
*      return code
*      HPSS_API_BUSY_DELAY       - Used to control the number of seconds
*
*      to delay between retry attempts.
*      HPSS_API_BUSY_RETRIES     - Used to control the number of retries
*
*      to be performed when a request fails
*      because the Core Server does not
*      currently have an available thread
*      to handle the request
*      HPSS_API_TOTAL_DELAY      - Used to control the number of seconds
*
*      to continue retrying a request
*      HPSS_API_LIMITED_RETRIES  - Used to control the number of retry
*
*      attempts before a limited retry error
*      operation fails
*      HPSS_API_DMAP_WRITE_UPDATES - Used to control the frequency of the
*
*      cache invalidates that are issued to
*      the DMAP file system while writing
*      to a file that is mirrored in HPSS
*      HPSS_API_REUSE_CONNECTIONS - Used to control whether TCP/IP
*
*      connections are to left open as long
*      as an HPSS file is open or are closed
*      after each read or write operation.
*      HPSS_API_USE_PORT_RANGE    - Used to control whether HPSS Movers
*
*      should the configured port range
*      when making TCP/IP connection for
*      read or write operations for the
client
*
*      HPSS_API_RETRY_STAGE_INP  - Used to control whether retries are
*
*      attempted on when trying to open
files
*
*      in a Class of Service that is
*      configured for background staging on
*      open
*      HPSS_API_DISABLE_CROSS_REALM - Used to control cross-realm traversal
*
*      HPSS_API_DISABLE_JUNCTIONS - Used to control junction traversal
*
*      HPSS_API_SITE_NAME         - Used to control which HPSS site
*
*      the client is connected
*      HPSS_API_AUTHN_MECH        - Used to control the select of an
*
*      authentication mechanism
*      HPSS_API_RPC_PROT_LEVEL     - Used to control the select of an
*
*      RPC protection level
*
*
*****
*
*/
{ "AIXTHREAD_COND_DEBUG",          "OFF",          NULL},
{ "HPSS_API_HOSTNAME",            "${HPSS_HOST}", NULL},
{ "HPSS_API_DESC_NAME",           "Client_Application", NULL},
{ "HPSS_API_DEBUG_PATH",          "stdout",       NULL},
{ "HPSS_API_MAX_CONN",            "0",            NULL},
{ "HPSS_API_DEBUG",               "0",            NULL},
{ "HPSS_API_RETRIES",             "4",            NULL},
{ "HPSS_API_BUSY_DELAY",          "15",           NULL},
{ "HPSS_API_BUSY_RETRIES",        "3",            NULL},

```

```

        { "HPSS_API_TOTAL_DELAY",          "0",          NULL},
        { "HPSS_API_LIMITED_RETRIES",     "1",          NULL},
        { "HPSS_API_DMAP_WRITE_UPDATES",  "20",        NULL},
        { "HPSS_API_REUSE_CONNECTIONS",   "0",          NULL},
        { "HPSS_API_USE_PORT_RANGE",      "0",          NULL},
        { "HPSS_API_RETRY_STAGE_INP",     "1",          NULL},
        { "HPSS_API_DISABLE_CROSS_REALM", "0",          NULL},
        { "HPSS_API_DISABLE_JUNCTIONS",   "0",          NULL},
        { "HPSS_API_SITE_NAME",           "${HPSS_SITE_NAME}", NULL},
        { "HPSS_API_AUTHN_MECH",          "${HPSS_CLIENT_AUTHN_MECH}",
          NULL},
        { "HPSS_API_RPC_PROT_LEVEL",      "${HPSS_RPC_PROT_LEVEL}",
          NULL},
        { "HPSS_API_SAN3P",               "1",          NULL},
/*
*****
*
* HDM Specific
*
*   HPSS_HDM_CAT_NAME - HPSS HDM message catalog full pathname
*   HPSS_HDM_SHMEM_KEY - The HDM's shared memory key (default 3789)
*   HPSS_HDM_SERVER_ID - The HDM's Server ID (default 1)
*****
*/
        { "HPSS_HDM_CAT_NAME",            "${HPSS_PATH_MSG}/hdm.cat",
          NULL},
        { "HPSS_HDM_SERVER_ID",          "1",          NULL},
        { "HPSS_HDM_SHMEM_KEY",          "3789",       NULL},
        { "HPSS_PATH_HDM",                "${HPSS_PATH_VAR}/hdm/hdm1", NULL},
/*
*****
*
* DMAP Extended Transaction Specific
*
*   HPSS_DMAP_TRANS_NUM - The number of concurrent extended
*                         transactions allowed
*   HPSS_DMAP_TRANS_STALE_SECS - The number of seconds before an
extended
*                         transaction is a candidate for being
*                         aborted
*   HPSS_DMAP_TRANS_CLEANUP_SECS - The number of seconds to sleep
between
*                         searches for stale transactions
*****
*/
        { "HPSS_DMAP_TRANS_NUM",          "10",         NULL},
        { "HPSS_DMAP_TRANS_STALE_SECS",   "30",         NULL},
        { "HPSS_DMAP_TRANS_CLEANUP_SECS", "10",         NULL},
/*
*****
*
* LOG Specific
*
*   HPSSLOG_SHMKEY - The HPSS logging shared memory key
*                   (default 2801)
*   HPSSLOG_SHMID - The HPSS logging shared memory id
*   HPSSLOG_HOSTNAME - Host name to use to connect to the log
client
*   HPSSLOG - HPSS log message output destination
*   HPSSLOGGER - HPSS logger output destination

```



```

*          [ stdout | syslog ]
*      HPSS_CONFIG_CAT_NAME - HPSS config message catalog full pathname
*      HPSS_INFRA_LOG_TYPES - Default types of infrastructure messages
*                          to be logged
*      HPSS_INFRA_LOG_CONF  - The infrastructure logging configuration
file
*      HPSS_MASTER_CAT_NAME - HPSS master message catalog full pathname
*
*****
*
*/
    { "HPSSLOG_SHMKEY",          "2801",          NULL},
    { "HPSSLOG_SHMID",         NULL,           NULL},
    { "HPSSLOG_HOSTNAME",     "${HPSS_HOST}",  NULL},
    { "HPSSLOGGER",           "",             NULL},
    { "HPSSLOG",               "",             NULL},
    { "HPSS_CONFIG_CAT_NAME",  "${HPSS_PATH_MSG}/config.cat",
      NULL},
    { "HPSS_INFRA_LOG_TYPES",  "CS_ALARM:CS_EVENT",  NULL},
    { "HPSS_INFRA_LOG_CONF",   "${HPSS_PATH_TMP}/$.log.conf",
      NULL},
    { "HPSS_MASTER_CAT_NAME",  "${HPSS_PATH_MSG}/hpss.cat",
      NULL},
/*
*****
*
* FTPD Specific
*
*      HPSS_PATH_FTP          - FTP daemon ./etc pathname
*      HPSS_FTPD_CONTROL_PORT - FTP daemon control default port ID
*      HPSS_FTP_RESERVED     - FTP reserved port IDs
*      HPSS_FTP_BLOCK_SIZE   - FTP block size
*****
*
*/
    { "HPSS_PATH_FTP",          "${HPSS_PATH_VAR}/ftp",  NULL},
    { "HPSS_FTPD_CONTROL_PORT", "4021",                 NULL},
    { "HPSS_FTP_RESERVED",     "1025",                 NULL},
    { "HPSS_FTP_BLOCK_SIZE",   "4",                    NULL},
/*
*****
*
* Security Registry & Service Location Specific
*
*      HPSS_SEC_REALM_NAME    - Default security realm name
*      HPSS_SITE_NAME        - Default HPSS site name
*      IEEE_802_FILE         - Location of file containing the MAC
address
*      HPSS_AUTH_SERVICE_CONF - File for valid authentication mechanisms
*      HPSS_AUTHZ_SERVICE_CONF - File for valid authorization mechanisms
*      HPSS_SEC_EP_CONF      - File containing the local endpoints
*      HPSS_KRB5_AUTHN_MECH  - Kerberos authentication mechanism
*      HPSS_KRB5_KEYTAB_FILE - The path for the kerberos keytab file
*      HPSS_UNIX_AUTHN_MECH  - HPSS Unix authentication mechanism
*      HPSS_UNIX_KEYTAB_FILE - The path for the HPSS Unix keytab file
*      HPSS_PRIMARY_AUTHN_MECH - The primary authentication mechanism to
use
*      HPSS_CLIENT_AUTHN_MECH - The client authentication method to use
*      HPSS_SEC_SITE_CONF    - File containing connect information for
*                          local security registry and location
*                          service
*
*      HPSS_AUTHN_TYPES      - Supported authentication types

```

```

*      HPSS_AUTHZ_TYPES          - Supported authorization types
*      HPSS_SITE_LOCATION        - Site Location
*      KRB5_INSTALL_PATH         - Kerberos installation path
*                                no default - platform dependent
*      KRB5_KDC_DIR              - Kerberos directory containing local
config
*                                files for KDC
*      KRB5_KDC_HOST             - Host for Kerberos KDC (just used by
mkhpss)
*      HPSS_LDAP_URL             - If set and non-empty, specifies the URL of the LDAP server that
*                                the hpss ldap admin tool should connect to by default.
*      HPSS_LDAP_SSL_KEYDB       - If set and non-empty, specifies the path to the SSL key db
*                                to use for SSL and indicates that SSL should be used to
*                                communicate with LDAP servers.  If this is used, it is
*                                assumed that that a corresponding password stash file
*                                exists as well.  This is the SSL stash (.sth) file, not
*                                the HPSS stash file used for SIMPLE LDAP binding.
*
*                                Do not set a default value; unset means something.
*
*      HPSS_LDAP_BIND_TYPE       - Specifies the type of binding that should be done with LDAP
servers.
*
*                                This is independent of whether SSL is used in the connection to
*                                the LDAP server.  You can still have encrypted communication if
you
*                                use GSSAPI, for example:
*                                - NONE - no bind is done; unauthenticated access
*                                - SIMPLE - simple (i.e. dn/password binding) determined by the
*                                    settings of the following:
*                                    - if HPSS_LDAP_BIND_ARG is set, it specifies the path to a
*                                      stash file containing the dn and password to use; see
*                                      ldap_stash.template for an example.
*                                      if not set, an error is generated.
*                                - GSSAPI - Kerberos binding via SASL.
*                                - (other) - an error is generated
*
*                                Do not set a default value; unset means something.
*
*      HPSS_LDAP_BIND_ARG        - Specifies further data necessary to complete a bind.
*                                Interpretation is based on the setting of
*                                HPSS_LDAP_BIND_TYPE (which see).
*
*                                Do not set a default value; unset means something.
*
*****
*/
{ "HPSS_SEC_REALM_NAME",          "%L",          NULL},
{ "HPSS_SITE_NAME",              "%H",          NULL},
{ "HPSS_SEC_REALM_ADMIN",        "admin/admin", NULL},
{ "HPSS_KRB5_AUTHN_MECH",        "krb5",        NULL},
{ "HPSS_KRB5_KEYTAB_FILE",       "${HPSS_PATH_ETC}/hpss.keytab",
  NULL},
{ "HPSS_UNIX_AUTHN_MECH",        "unix",        NULL},
{ "HPSS_UNIX_KEYTAB_FILE",       "$
{HPSS_PATH_ETC}/hpss.unix.keytab",
  NULL},
{ "HPSS_PRIMARY_AUTHN_MECH",     "${HPSS_KRB5_AUTHN_MECH}",

```

```

        NULL},
    { "HPSS_PRIMARY_AUTHENTICATOR", "${HPSS_KRB5_KEYTAB_FILE}",
      NULL},
    { "HPSS_CLIENT_AUTHN_MECH", "${HPSS_PRIMARY_AUTHN_MECH}",
      NULL},
    { "IEEE_802_FILE", "${HPSS_PATH_ETC}/ieee_802_addr",
      NULL},
    { "HPSS_AUTH_SERVICE_CONF", "${HPSS_PATH_ETC}/auth.conf",
      NULL},
    { "HPSS_AUTHZ_SERVICE_CONF", "${HPSS_PATH_ETC}/authz.conf",
      NULL},
    { "HPSS_SEC_EP_CONF", "${HPSS_PATH_ETC}/ep.conf",
      NULL},
    { "HPSS_SEC_SITE_CONF", "${HPSS_PATH_ETC}/site.conf",
      NULL},
    { "KRB5_CONF", "${HPSS_PATH_SLASH_ETC}/krb5.conf",
      NULL},
    { "HPSS_AUTHN_TYPES", "krb5,unix", NULL},
    { "HPSS_AUTHZ_TYPES", "ldap,unix", NULL},
    { "HPSS_SITE_LOCATION", "USA", NULL},
    { "KRB5_INSTALL_PATH", "", NULL},
    { "KRB5_KDC_DIR", "${HPSS_PATH_VAR}/krb5kdc",
      NULL},
    { "KRB5_KDC_HOST", "", NULL},
    { "HPSS_LDAP_URL", "", NULL},
    { "HPSS_LDAP_SSL_KEYDB", "", NULL},
    { "HPSS_LDAP_BIND_TYPE", "", NULL},
    { "HPSS_LDAP_BIND_ARG", "", NULL},
    { "HPSS_UNIX_AUTH_PASSWD", "/var/hpss/etc/passwd", NULL},
    { "HPSS_UNIX_AUTH_SHADOW", "/var/hpss/etc/shadow", NULL},
    { "HPSS_UNIX_AUTH_GROUP", "/var/hpss/etc/group", NULL},
    { "HPSS_UNIX_USE_SYSTEM_COMMANDS", "TRUE", NULL},
}

/*
*****
*
* RPC Specific
*
*   HPSS_RPC_PORT_RANGE - Range of TCP/IP ports to use for RPCs
*   HPSS_RPC_SOCKETBUF_SZ - The RPC socket buffer size to be used
*   HPSS_RPC_SOCKETIO_SZ - The RPC socket I/O size to be used
*   HPSS_RPC_PROG_NUM_RANGE - The range for RPC program numbers
*   HPSS_RPC_PROG_NUM_HDM - The HDM's well-known program number
*   HPSS_RPC_PROT_LEVEL - Default RPC protection level to be used
*
*****
*/
{ "HPSS_RPC_PORT_RANGE", NULL, NULL},
{ "HPSS_RPC_SOCKETBUF_SZ", NULL, NULL},
{ "HPSS_RPC_SOCKETIO_SZ", NULL, NULL},
{ "HPSS_RPC_PROG_NUM_RANGE", "0x20000000-0x20000200", NULL},
{ "HPSS_RPC_PROG_NUM_HDM", "0x20000200", NULL},
{ "HPSS_RPC_PROT_LEVEL", "connect", NULL},
}

/*
*****
*
* Installation & Miscellaneous
*
*   HPSS_PATH_ADM - Path where administrative files are
placed

```

```

*      HPSS_PATH_CORE          - Path where subsystem core files are
placed
*      HPSS_PATH_TMP           - Path where temporary files are placed
*      HPSS_PATH_ETC           - Path where runtime config files are
placed
*      HPSS_ENV_CONF           - The path to the environment override
file
*      HPSS_PTHREAD_STACK      - Stack size for HPSS pthreads
*****
*
*/
    { "HPSS_PATH_ADM",          "${HPSS_PATH_VAR}/adm", NULL},
    { "HPSS_PATH_CORE",        "${HPSS_PATH_VAR}/core", NULL},
    { "HPSS_PATH_TMP",         "${HPSS_PATH_VAR}/tmp",  NULL},
    { "HPSS_PATH_ETC",         "${HPSS_PATH_VAR}/etc",  NULL},
    { "HPSS_PATH_CRED",        "${HPSS_PATH_VAR}/cred", NULL},
    { "HPSS_ENV_CONF",         "${HPSS_PATH_ETC}/env.conf",
      NULL},
    { "HPSS_PTHREAD_STACK",    "131072",                NULL},
    { NULL,                    NULL,                    NULL}
};

```

Appendix F. */var/hpss* files

The */var/hpss* directory tree is the default location of a number of HPSS configuration files, log files, and other files needed by the servers.

The directories and configuration files can be created with the *mkhps* utility or hand created. Be very careful when using *mkhps* utility as selecting the wrong option can damage the already partially configured HPSS 6.2 system. The log files and other files are created by the servers.

The directories in */var/hpss* include:

acct. The usual directory to hold accounting report files and checkpoint files. This location is specified in the Accounting Policy. The administrator must regularly remove unneeded reports to prevent the */var/hpss* filesystem from filling.

adm. This directory contains one log file and one subdirectory:

- **hpssd.failed_server.** A file of log entries for server startups and terminations. Created and maintained by the startup daemon.
- **core.** The default directory where HPSS servers put “core” files if they terminate abnormally. The system administrator must clean out old core files regularly to avoid filling up */var/hpss*.

cred. The default directory where Kerberos credential files for HPSS servers are placed.

doc. The default directory where the HPSS documentation is installed.

etc. The default directory where many UNIX configuration files are placed. These files include:

- **HPSS.conf.** An ftp configuration file. See Appendix D: *HPSS.conf Configuration File*.
- **auth.conf.** Created by *mkhps*. Lists HPSS GAS libraries and functions for performing initialization of authentication manager: Kerberos, Unix, or GSI.
- **authz.conf.** Created by *mkhps*. Lists HPSS SEC libraries and functions for performing initialization of authorization manager: LDAP or Unix.
- **env.conf.** Created as empty file by *mkhps*. Contains site specific environment variables for HPSS. This file is new in HPSS 6.2. Upon startup, utilities and servers in HPSS automatically source this file for necessary environment variables.
- **ep.conf.** Created by *hpss_bld_ep* program in *HPSS_ROOT/config*. Contains the UUID endpoints for clients to contact the HPSS sites.
- **hpss.keytab.** Created by *mkhps*. Contains the username and password for HPSS servers to use for Kerberos authentication with a keytab file.
- **hpss.unix.keytab.** Created by the *hpss_unix_keytab* program. Contains the username and password for HPSS servers to use for Unix authentication with a keytab file.
- **mm.keytab.** Created by *mkhps*. The keytab for user *hpss*. Used by utilities which read the DB2 databases.
- **ieee_802_addr.** Created by *mkhps*. Contains the MAC address for the primary network interface for the machine.
- **site.conf.** Created by *mkhps*. Contains the site name, realm name, realm Id,

authorization mechanism and authorization URL to utilize.

- **rc.db2.** Created by mkhpss. Script for starting DB2.
- **rc.krb.** Created by mkhpss. Script for starting the Kerberos servers.
- **passwd.** Created by mkhpss. A local HPSS-only password file for use with Unix authentication and authorization. Optionally, the system password file can be used instead.
- **group.** Created by mkhpss. A local HPSS-only group file for use with Unix authentication and authorization. Optionally, the system group file can be used instead.
- **unix.master.key.** Created by mkhpss. The file can also be generated by running the `hpss_unix_keygen` utility. It contains a hexadecimal key in the format `0x#####`
`0x#####`. The key is used by HPSS during Unix authentication to provide an extra layer of protection to the entries in the `hpss.unix.keytab` file. Passwords are wrapped/unwrapped using the master key if the `GAS_UNIX_CRED_MASTER_KEY` flag is set. The `unix.master.key` file should only exist on the core server machine and should be adequately protected.

ftp. Files for ftp configuration. See Section 13.2: *FTP Daemon Configuration* in the *HPSS Management Guide* for more information.

gk. The recommended directory for the site policy configuration file used by the Gatekeeper if the Gatekeeper is configured to do Gatekeeping Services. The file is usually named `gksitepolicy`. It is created by the system administrator.

hdm. The directory to hold files for XFS configuration.

hpssdb. The directory to hold the DB2 instance configuration information and 'CFG' database tables.

krb5kdc. The directory for Kerberos configuration files. These include:

- **kadm5.keytab.** The keytab for the Kerberos administrative user
- **kdc.conf.** Kerberos configuration file. See the Kerberos documentation.

log. The default directory where the Log Daemon creates two central log files, `logfile01` and `logfile02`. In this same directory, the Log Client will continually write a circular ASCII log file, `local.log`.

mpps. The directory in which the MPS places its migration/purge reports, if it is configured to produce these reports. Reports are generated every 24 hours. The system administrator will need to remove these reports regularly when no longer needed to avoid filling up the `/var/hpss` file system.

ssm. The usual directory for ssm configuration files. These include:

- A file to store alarms and events if SSM is configured to buffer alarm and event messages in a disk file (as opposed to keeping them in memory). This pathname is defined by the `HPSS_SSM_ALARMS` environment variable.
- An optional subdirectory to hold keytabs for `hpssadm` users on the system. The directory and keytab files are created by the system administrator.
- **login.conf.** Contains information required by SSM authentication. Created by `hpssuser`.

See Chapter 3: *Using SSM* of the *HPSS Management Guide* for details.

tmp. The `/var/hpss/tmp` is the default directory where the Startup Daemon creates a lock file for each

of the HPSS servers it brought up in the node. HPSS may also write diagnostic log files and disk allocation maps in this directory, when configured to do so. The lock files are very small, but the log files and maps can be very large.